Context-Aware Cognitive Radios Learning from Data Using Machine Learning Techniques

Baban, Shaswar Tharwat Mohammed

Awarding institution:
King's College London

Context-Aware Cognitive Radios Learning from Data Using Machine Learning Techniques

Shaswar Baban

School of Natural & Mathematical Sciences

King's College London

This dissertation is submitted for the degree of

Doctor of Philosophy

June 2016

I would like to dedicate this thesis to my loving parents,

Sawsan and Tharwat Baban ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgments. This dissertation contains fewer than 100,000 words including appendices, bibliography, footnotes, tables and equations.

Shaswar Baban

June 2016

Acknowledgements

First and foremost, I would like to thank my family in general and my parents in particular for their constant and unconditional support all throughout.

I would like to acknowledge my primary supervisor, Hamid Aghvami, for his guidance and sincere personal advice - I would always remember the deep conversation on the train back to London from Paris. Likewise, my secondary supervisor, Fatin Said, who was also supportive and ever encouraging. Thanks to all my friends at the Centre for Telecommunications Research at King's College London. Thanks to the EU FP7-ICT project ACROPOLIS, which funded my research visits, conferences, and summer schools.

Toward the end of my PhD, I joined the Cognitive Radio Working Group at Wireless Innovations Forum where it was a pleasure to work on Big Data and Cognitive Radio concepts with James Neel, Ihsan Akbar, Charles Sheehe, Neal Mellen, Peter Cook, Bob Schutz and Daniel Davasivatham.

Other people whom I exchanged ideas with at one stage or another, many thanks to you all: David Silver at UCL who allowed me to take his course on Reinforcement Learning; Jose Llarena and Matthew Howard for the discussions on Q-Learning and convergence issues; Thanks to Maria Fox, Mohammad Shikh-Bahaei, Panos Kosmas, Trever Pearce, and many others at KCL for their support when it came to tedious admin issues; all the amazing friends I made at Goodenough College, for the interesting dinner time chats on quantum

physics, global economics, politics, genetics, and what not, it was a pleasure to know you personally from close. Thanks to GC in general, for helping me financially and letting me be part of this great place, that is.

Last but not least, it is also worth mentioning all openly available online digital platforms such as Wikipedia, coursera, edX, and many others that helped me further understand and learn about the latest algorithms free of charge. Thank you *Internet* , in general; you have been my main educator.

Abstract

Wired or wireless, connectivity has been a vital commodity of life and more so recently in the realm of information age. Those who have access to faster, more reliable and more ubiquitous connectivity—put simply, those who are "better connected"—will have significant advantages in commerce, research and a host of other arenas. In regards to wireless communications, due to the explosion in demand for higher capacity networks, availability of free spectrum resources have become increasingly scarce. The UHF spectrum band in particular, due to its excellent electromagnetic properties, has been reported as inefficiently used and congested by many spectrum regulators of the world. This spectrum resource scarcity issue combined with the ongoing research and development for more intelligent, autonomous and self-aware radio communication led to a vast amount of research on the concept of Cognitive Radio.

This thesis researches the learning unit of cognitive radios. The learning unit is responsible for processing information and autonomous decision making. In particular, the research is focused on the extraction and usage of contextual information from the radio environment (e.g. Radio Access Technology type, channel access pattern learning/recognition) and how such information could be exploited to improve the performance of the cognitive radio. The key metrics discussed will be based on information extraction under noise, channel blocking and interference reduction to primary users.

We present a set of novel works involving Machine Learning, which is a branch of Artificial Intelligence. New implementation and use cases of state-of-the-art machine learning algorithms are presented that learn from real-life data. In a testbed setup we program software defined radios to recognize different Radio Access Technologies and their channel access patterns. The main technique used in the majority of the thesis is Artificial Neural Networks, concretely: Multi-Layer Perceptron Neural Nets, Self-Organizing Neural Nets, and Deep Auto-Encoders. In some of the works these neural network architectures have been combined in a novel way with Support Vector Machines, and Reinforcement Learning algorithms for channel classification and access.

In this thesis we show that it is possible to achieve 95% correct classification at -25 dB among three different radio access technologies, namely, DVB-T, WCDMA and IEEE 802.11a, where, consequently, we can reason over the outcome of this classification to differentiate between primary and secondary transmissions. We also show that, through the use of the proposed auto-encoder approximate Q-learning technique, such context-aware cognitive radio can achieve better key performance metrics in dynamic spectrum access as compared to previously researched Q-learning algorithms.

Table of Contents

List of Figures

List of Tables

List of Acronyms

Acronyms / Abbreviations

AE Auto-Encoder

AI Artificial Intelligence

AppQL Approximate Q-Learning

AWGN Additive White Gaussian Noise

BBU Base Band Unit

BER Bit Error Rate

BS Base Station

CAPEX Capital Expenditure

CDMA Code Division Multiple Access

CNN Convolutional Neural Network

CPU Central Processing Unit

CR Cognitive Radio

CRN Cognitive Radio Network

CRWG Cognitive Radio Working-Group

CSMA Carrier Sense Multiple Access

CUD Channel Utilisation Diversity

DAB Digital Audio Broadcasting

DBN Deep Belief Network

DECT Digital Enhanced Cordless Telecommunications

DNN Deep Neural Network

DSA Dynamic Spectrum Access

DSSS Discrete Sequence Spread Spectrum

DVB-T Digital Video Broadcasting – Terrestrial

ETSI European Telecommunications Standards Institute

EU European Union

FCC Federal Communications Commission

FHSS Frequency Hopping Spread Spectrum

GDP Gross Domestic Product

3GPP 3rd Generation Partnership Project

GPU Graphical Processing Unit

GQ Greedy Q-Learning

GSM Groupe Spécial Mobile

GSMA GSM Association

HDFS Hadoop File System

HIL Human-In-the-Loop

ICT Information communication Technology

IEEE Institute of Electrical and Electronic Engineers

IMT International Mobile Telecommunication

IP Internet Protocol

ISM Industrial, Scientific and Medical (radio band)

IT Information Technology

ITU International Telecommunications Union

KL Kullback–Leibler

LAN Local Area Network

LTE Long Term Evolution

LVQ Linear Vector Quantization

MAC Medium Access Control

MDP Markov Decision Process

MLP Multilayer Perceptron

MSE Minimum Square Error

NFV Network Function Virtualization

NN Neural Network

OECD Organisation for Economic Co-operation and Development

OFCOM The Office of Communication

OFDM Orthogonal Frequency Division Multiplexing

OFDMA Orthogonal Frequency Division Multiple access

OPEX Operational Expenditure

PC Personal Computer

PDA Personal Digital Assistant

PSD Power Spectral Density

PU Primary User

QL Q-Learning

QoE Quality-of-Experience

RAM Random Access Memory

CRAN Cloud Radio Access Network

RAT Radio Access Technology

RBF Radial Basis Function

RBM Recursive Boltzmann Machine

REM Radio Environment Map

RF Radio Frequency

RL Reinforcement Learning

RM Robins Monro

RRH Remote Radio Head

RRU Remote Radio Unit

RTT Round Trip Time

SAE Sparse Auto-Encoder

SDN Software Defined Network

SDR Software Defined Radio

SINR Signal-to-Interference-plus-Noise Ratio

SNR Signal-to-Noise Ratio

SOM Self Organising Map

SON Self-Organising Network

SSAE Stacked Sparse Auto-Encoder

SU Secondary User

SVM Support Vector Machine

TD Temporal Difference

TL Transfer Learning

TTI Transmission Time Interval

TU Typical Urban

TVWS TV White Space

UHF Ultra High Frequency

UMTS Universal Mobile Telecommunications System

USRP Universal Software Radio Peripherals

WCDMA Wideband Code Division Multiple Access

WRC World Radio Conference

WSD White Space Device

WSDB White Space Database

Chapter 1

Introduction

1.1 Introduction

For the past decades wireless mobile communications has taken up a fundamental role in bringing humanity together. The advent of telecommunication services have induced a boost to our daily life activities. Nonetheless, when the usage of such technologies rises to facilitate lives of masses of people, the resources upon which they rely on ought to scale accordingly. One of the fundamental resources that wireless communications needs to manage efficiently is the radio spectrum. The radio spectrum is a part of the electromagnetic spectrum ranging from 3 kHz to 300 GHz, portions of which are allocated by spectrum regulators around the world for various uses, such as mobile broadband, television, satellite, radar and so on.

Recently, however, various national and international spectrum regulators have reported [3–5] that the sweet spot of the radio spectrum—or in other words, the UHF band that is situated between 300 and 3000 MHz—could not accommodate further capacity, since most of this spectrum has been already allocated,

and thus experiencing what was dubbed as the *spectrum crunch*. Indeed, just like any natural resource the radio spectrum is also limited if interference protection ratios between communication devices are to be considered.

There are several factors that underpin the spectrum crunch, some of which are:

- Increase of demand for ubiquitous access to information: it is predicted that connected devices' traffic growth will be 11-times in 2013–2018 [6] and monthly global mobile data traffic will surpass 24.3 exabytes by 2019 [1].
- Demand for higher connection speeds: in 2019 average mobile network connection speeds will be more than two-folds reaching 4Mbps from 1.7Mbps in 2014 [1].
- Connecting the last billion people to the Internet is a more challenging task than what we have achieved. Most of the connected areas are urban and suburban areas, however, the rest come from rural regions. Due to the nature of this challenge, farther traveling signals are preferred to be used in such areas of sparse population. Again, for this purpose, the sub-gigahertz band contains the most desirable frequencies. Yet, because of the mentioned desirable features of the sub-gigahertz spectrum, this band is already congested by various public and private communications systems.

Conventionally, radio spectrum licensing has been managed in a command-and-control manner, as shown in the left side of Figure 1.1. In this regard, owners of a specific chunk of spectrum have total usage rights in specific parts of a country according to the license; others are not allowed to transmit in the license holders' allocated spectrum and region of the country where the license

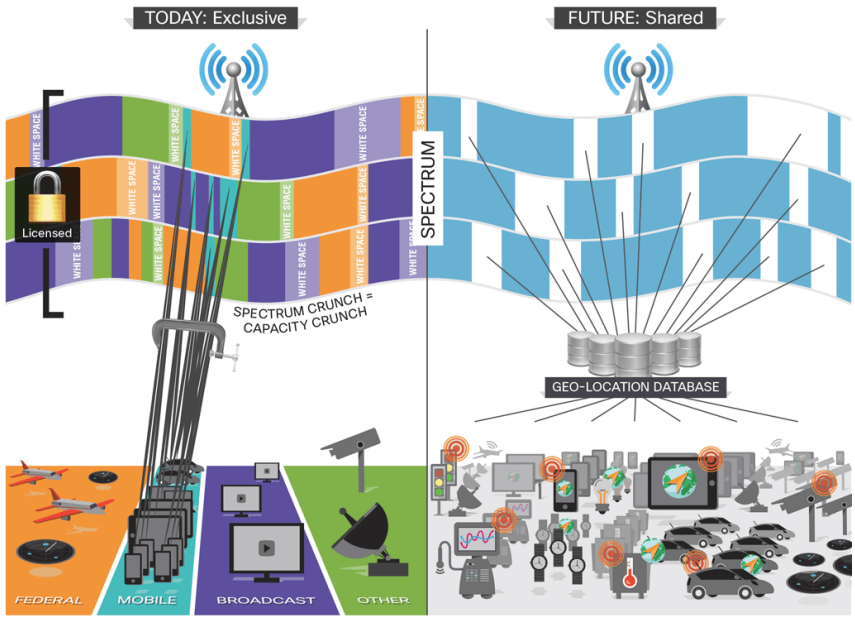


Fig. 1.1 The Past and the Future - Dynamic Spectrum Access – directly reproduced from [6]

is dictated to. According to an FCC Spectrum Policy Task Force Report published in 2002 [3], this static allocation of portions of spectrum was found to result in a very-low spectrum utilization level. For instance, the report indicates that there are different temporal and spatial variations of the utilized spectrum, ranging from 15% to 85% [7].

Therefore, more efficient utilization of such scarce resources becomes a fundamental issue as we further advance and rely more on wireless communication to deliver our messages.

1.1.1 What is Cognitive Radio?

Joseph Mitolla III in 1999 proposed the Cognitive Radio (CR) concept [8–10] as an emerging wireless networking technology, describing it as:

> "The point in which wireless personal digital assistants (PDAs) and the related networks are sufficiently computationally intelligent

> about radio resources and related computer-to-computer communications to detect user communications needs as a function of use context, and to provide radio resources and wireless services most appropriate to those needs."

Essentially, the vision of Cognitive Radios was to be implemented as a Software Defined Radio (SDR) [11]. Empowered by the flexibility of SDRs, that enable CRs to be (re)configured in software, and ever increasing demand in wireless communications, CR became one of the key topics of research in the first decade of the new millennium. One of the prominent applications of the CR concept is Dynamic Spectrum Access (DSA) [12]. DSA has been researched vigorously in the past decade as well principally because of the spectrum crunch as discussed above. There are different ways to dynamically access spectrum. However, in general, it is a technique which incorporates getting knowledge about the available spectrum, e.g. scanning the radio environment for transmission opportunities, deciding which channel to access in case there where more than one empty channel, then dynamically switching its communication seamlessly to the new channel. A diagram of the Cognition Cycle, which is originally taken from agent modeling in computer science, is shown in Figure 1.2.

In this context, sensing is a key enabler to the cognitive radios' functioning. As the first unit in the cognitive radio's cognition cycle, it executes the function of signal detection. However, the premise with CR is that, knowing the channel fluctuations—be it in terms of predicting the channel or having a separate apparatus to sense the channel—is not enough information for the CR to behave according to the regulations dictated by international regulators such as the FCC [3, 13] or OFCOM [4, 5]. What is common and prominent among rules set out by most regulatory bodies is that transmissions of *secondary* users

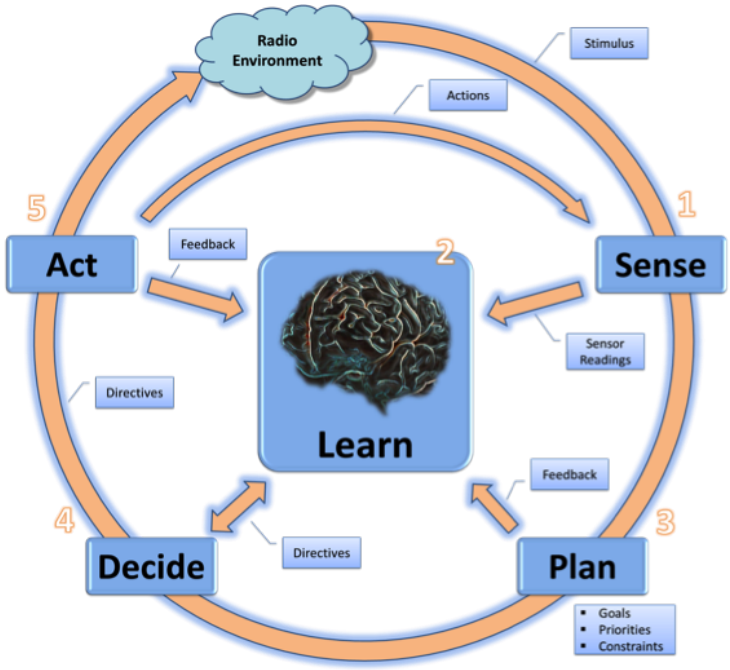


Fig. 1.2 Cognition Cycle [8]

who do not own the unrestricted right of spectrum usage in a particular band, must not interfere with the communication of the incumbent or in other words *primary* users.

In this thesis we will be considering scenarios where cognitive radios are allowed to access primary bands whenever they are vacated, on the condition of not causing perceivable interference. In this context, it is very important for any secondary cognitive user to be able to recognize the type of the ongoing radio communication, be it primary or secondary, in a particular band of interest, and infer the priority of usage of that band accordingly.

1.2 What are the research challenges?

The main problem this thesis is attempting to solve at large is capacity improvement of wireless communication networks. However, as discussed in Section 1.1.1, this could be done indirectly through more spectrum-efficient techniques such as DSA.

The challenges that we are facing here, due to operation as a CR, are various:

- DSA: Considering the DSA technique, it is not enough to detect a band and label it as occupied or not occupied, since, in the case of band occupancy by a secondary user, other secondary users should not lose their chance to contend. Assuming that certain Radio Access Technology (RAT) types identify whether a user is a primary or a secondary user (e.g. if detected Wi-Fi then Secondary, otherwise if UMTS (Universal Mobile Telecommunications System) was detected then primary), we can therefore conclude that, the type of the RAT occupying a spectrum band is essential for others to know whether they are allowed to access a band or not when they are found to be unoccupied.

- Collaboration to Cooperate: Considering the scarce spectrum again, these secondary networks may want to collaborate to cooperate and share the spectrum band that was found to be unoccupied at an instant. However, if these secondary networks use different RATs, it would be difficult to collaborate, which is simply because they would not be able to understand each other's protocols, unless they communicated using the same RAT.

- Cognitive Vertical Handover: The proliferation of different types of RATs that has given rise to the concept of Heterogeneous Networks or HetNets, consequently lead to the requirement of seamless handover of communica-

tion sessions between RATs. This type of handover is termed as Vertical or Inter-RAT Handover. In a DSA enabled CR HetNet scenario, a CR may need to be able to classify different RATs, so that it can infer the handover procedures, and reconfigure its RAT to that of the network that it intends to handover to.

- Radio Environmental Map (REM): A REM is a database that contains knowledge about the radio environment. The goal of such a knowledge-base is to aid cognitive (or even non-cognitive) radios to optimize their operation through additional information analyses at a REM database such as: geolocation, available services and networks, regulatory policies, information regarding nearby radios, and so on. In the same sense, the type of the access technology used by neighboring radios has an important role, such as in facilitating secondary-secondary spectrum sharing, where SUs identifying each other's RATs results in re-configuring their RATs so that they understand each other.

All of the above bullet points call for the necessity of a technique for CRs to be able to recognize different types of current RATs, and learn to identify future RATs automatically.

Furthermore, a technique as such opens doors to many other questions, some of which are listed below. Assuming that we are able to differentiate between PUs and SUs transmissions through this research,

- How do SUs using different RATs collaborate with each other? How fast can they converge to an optimum solution?
- How to share a temporary or spatial white space between SUs optimally?
- How can learned knowledge be transferred from one CR to another?

- How do CR interact with spectrum regulators? in other words, can regulators be assured that their regulations will be respected by secondary users?

The above and many more questions that we stumble upon in the forthcoming chapters, will guide the research herein.

1.2.1 Significance of the Problem

A large-scale pilot study on TV white-space spectrum availability in the UK under the supervision of OFCOM and other academic and industrial members ended in 2015 [4]. On the basis of this consultation, it was finally decided "to allow a new wireless technology access to the unused parts of the radio spectrum in the 470 to 790 MHz frequency band" [4]. In this sense, we can envision that in the near future, two or more secondary networks trying to co-exist in the sub-GHz band with primary users will be a very probable scenario. Therefore, we can safely predict a surge of new secondary networks, consisting of secondary users trying to access the new spectrum band under the rules of White Space Databases, regulated by OFCOM.

Another way to achieve capacity enhancement in wireless networks is to deploy smaller cells [14–16]. Naturally, since Local Area Network (LAN) Small Cells could be using different RATs, with their introduction the frequency of these inter-RAT or vertical handovers is likely to increase. Hence, it is imperative to come up with a solution that would identify different RATs available today, as well as new RATs that may be designed in the future.

All the above aside, there are no common frameworks for a Cognitive Radio's Learning Unit today that one can rely on, although many novel ideas have been proposed in literature in the past decade [17–20]. It is thus essential to come up

with a design for a Learning Unit that is able to integrate numerous different machine learning techniques studied by various researchers. These learning methods are usually diverse and are mainly used to optimize the operation of a specific unit of the cognitive radio cycle. The knowledge learned by these units is mostly used to improve the utility of these units separately. Through this research we attempt to fuse knowledge learned from different units of the cognitive cycle and infer more information from them in order to enhance the performance of a Cognitive Radio alone or a network of Cognitive Radios in totality.

1.3 An outline of the sections of the thesis

We start in Chapter 2 by conducting a literature review of CR and Machine Learning techniques used in previous research works. Later in the Chapter we also allude to future research challenges in our perspective for next generation communication systems. At the end of Chapter 2 we give a description of the research methodology and testbed setup used in the rest of the chapters.

The research done in this thesis can be generally divided into three different themes. In the first section of the this thesis, each chapter studies a different type of radio signal classification tested on various levels of artificially added noise. As the research develops, through making use of different machine learning techniques, each chapter attempts to better the classification performance at the cost of training more sophisticated learning algorithms. In this part of the thesis we are essentially researching mechanisms that make Cognitive Radios more context-aware. Context-awareness is, however, a broad subject. Therefore, in this thesis, we focus our efforts on Radio Access Technology recognition that would the priority context in channel access scheme. The

chapters that fall into this part are: Chapter 3, Chapter 4, and the first part of Chapter 5.

The second and main section or theme of the thesis discusses the role of RAT classification techniques in enabling more intelligent reduced primary-secondary collisions. Herein, a type of deep neural network called Stacked-Auto Encoders, along with a set of model-free and off-policy reinforcement learning algorithms are researched to provide an enhanced cognitive channel sensing and access scheme. Chapter 5 is responsible for this section and starts with the deep learning feature extraction in the beginning, followed by a detailed analysis of different Q–Learning algorithms.

Finally, in the future works section, some of our ongoing research in Big Data applications has been put forward. This section examines the parallels between Big Data problems and emerging CR and wireless applications, appropriate Big Data tools for RF domain problems, new RF-domain applications that could be enabled by Big Data, and other developments needed to enable CRs to maximize the capabilities offered by Big Data.

1.3.1 Summary of Chapters

Below is an abstract and summary of contributions of all forthcoming chapters:

Chapter 2: Background and Overview

We start in chapter 2 by reviewing the most prominent literature on cognitive radios, citing key articles and reports that laid the foundation of this line of research. Consequently, we also provide a technical background introduction to Matched Filtering which is the baseline technique used for comparison pur-

poses, and Neural Networks which forms the building blocks of most of the machine learning based classifiers proposed in this thesis. Toward the end of the chapter we emphasize on the data generation and collection process where a testbed layout is put forward.

Chapter 3: Radio Access Technology Classification for Cognitive Radio Networks using Multi-Layer Perceptron Neural Networks [21–23]

In spectrum bands where spectrum sharing is allowed by national regulators, radio access technology recognition is an important technique for reducing interference and facilitating cooperation among cognitive radios. Unlicensed users (secondaries) need to be able to differentiate between transmissions of licensed users (primaries) and other unlicensed users. Furthermore, secondaries should only free a band when the licensed primary user starts to transmit. In this regard, secondary users' transmission technology classification will have a vital role for coexistence/cooperation purposes in such shared spectrum bands. For the purpose of this work, a practical testbed made up of software defined radio transceivers and a set of computing units was put together. A classification multi-layer perceptron neural network was trained in a supervised learning method on a dataset of various radio access transmissions such as DVB-T, WCDMA and IEEE 802.11a WiFi. Testbed results demonstrate the efficiency of the classification, where The multi-layer perceptron architecture achieves 99% correct classification at SNR=-10 dB and 97% at SNR=-15 dB, while the baseline matched filter method only achieves 55% at SNR=-10 dB and 33% at SNR=-15 dB.

Chapter 4: Radio Environment Feature Extraction and Classification using Self-Organizing Feature Maps and Support Vector Machines [21, 24]

Radio access technology recognition can be an important technique to facilitate spectrum sharing, interference avoidance, and cooperation among cognitive radios. As one example among its very many possible uses and benefits, RAT recognition might allow secondary users to differentiate between the transmissions of primary users and other SUs, such that SUs might contend for a spectrum band fairly, only not transmitting when they detect the PUs as having started to transmit in the same band. In this work, a practical testbed made up of software defined radio transceivers and computing units has been assembled, and used to transmit and receive samples of representative RATs as was mentioned in Chapter 2. A Self-Organizing Feature Map with Support Vector Machine clustering and classification technique has been developed in a semi-supervised learning manner, to operate on these received samples. Finally, performance metrics have been presented showing 99% correct classification performance at -20 dB SNR and around 96% at -25 dB SNR, which demonstrates the efficiency of this technique over matched filter baseline methods which achieves only 33% at the above mentioned SNR levels.

Chapter 5: Radio Access Technology-Aware Cognitive Radio Spectrum Access: Using Deep Neural Nets and Reinforcement Learning [25]

With the advent of Deep Learning architectures, recently there has been a revolution in machine learning research in general. Powered by Graphical Processing Units (GPUs) and distributed computing, as well as novel train-

ing algorithms, deep neural network architectures have achieved significant increase in performance in both supervised and unsupervised training approaches. Side-by-side, there has also been improvements to Reinforcement Learning paradigm. In this chapter we combine a type of semi-supervised deep learning method, namely, Sparse Auto-Encoders, with modified version of Q-Learning, which is also a type of reinforcement learning. The goal of this work is to minimize collisions between primary and secondary users, where they are assumed to coexist in the sub-GHz spectrum band as discussed above. Through the use of extensive simulations we show that our combined auto-encoder approximated Q-learning approach has better feature extraction and channel access performance as compared to other Q-learning algorithms such as greedy, e-greedy and functional Q-learning. Results have been presented on various key performance indicators including, scalability, agility, mobility, convergence, diversity, complexity and learnability.

An overall conclusion of the research findings and subtleties are presented in Chapter 6 alongside a discussion on the research directions therein.

1.4 Contributions to Literature

The contributions by the author to the research community can be summarized as follows:

Conference Papers

1. **Shaswar Baban**, Daniel Denkoviski, Oliver Holland, Liljana Gavrilovska, Hamid Aghvami, “Radio Access Technology Classification for Overlay and Underlay Cognitive Radio Networks”, 2nd Annual Acropolis Work-

shop, SDR WInnComm'12 (2012). [23]

2. **Shaswar Baban**, Daniel Denkovski, Oliver D Holland, Liljana Gavrilovska, and Hamid Aghvami. "Radio Access Technology Classification for Cognitive Radio Networks" PIMRC'13 – Mobile and Wireless Networks (2013). [21]

3. Dagres, Ioannis, Andreas Polydoros, Daniel Denkovski, Valentin Rakovic, Vladimir Atanasovski, Liljana Gavrilovska, Krzysztof Cichon, Adrian Kliks, **Shaswar Baban**, and Oliver Holland. "An Integrated Platform for Source Detection, Identification and Localization with Applications to Cognitive-Radio." European Wireless 2013 (2013). [22]

4. **Shaswar Baban**, Oliver Holland, and Hamid Aghvami. "Wireless Standard classification in Cognitive Radio Networks using Self-Organizing Maps." ISWCS'13 CRAFT Workshop (2013). [24]

5. James Neel, Peter G. Cook, Ihsan Akbar, Neal Mellen, **Shaswar Baban**, Charles Sheehe, Bob Schutz, Daniel Devasirvatham, "IPA Volume 3: Cognitive Radio Context, WISDM, and Big RF", SDR-WInnComm – Technical Track: Top 10 Most Wanted Wireless Innovations, (2014). [26]

6. J. Neel, **Shaswar Baban**, P. Cook, N. Mellen, I. Akbar, D. Devasirvatham, C. Sheehe, B. Schutz, "Context-Aware Cognitive Radio for Automated Wireless System Management," SDR-WInnComm 14, March 11-13 (2014). [27]

7. J. Neel, **Shaswar Baban**, N. Mellen, I. Akbar, C. Sheehe, B. Schutz, P. Cook, "Big RF for Homeland Security Applications", IEEE HST Symposium, 2015. [28]

8. J. Neel, **Shaswar Baban**, P. Cook, I. Akbar, N. Mellen, C. Sheehe,

D. Devasirvatham, "Big RF for Management of Shared Spectrum Networks," Wireless Innovations' Forum on Wireless Communication Technologies and Software Defined Radio – WInnComm, San Diego, 2015. [29]

Journal Papers

1. **Shaswar Baban**, James Neel, Ihsan Akbar, Neal Mellen, Peter G. Cook, Charles Sheehe, Bob Schutz, Hamid Aghvami, "A Big Data Framework for Next Generation Wireless Networks: Big RF", Technical Report. [to be submitted in 2016]
2. **Shaswar Baban**, Hamid Aghvami, "RAT-Aware Cognitive Radio Spectrum Access: Using Deep Learning For Reduced Primary-Secondary Collision Rate", Technical Report. [to be submitted in 2016]

Green/Discussion Papers

1. techUK, "5G Innovation Opportunities", (2015). [30]

Contribution to EU ICT ACROPOLIS Project

1. Deliverable 10.3 Cooperative and Cognitive Networks: Architecture, Protocols and Evaluation. (2014)

Chapter 2

Background and Overview

2.1 Introduction

In the previous chapter we introduced the concept of Cognitive Radios and pointed out the main reasons that gave rise to the significant amount of research on this topic. In this chapter, we present an overview of the prominent works related to the use of machine learning techniques used in cognitive radio paradigm in general, and radio access technology identification and dynamic spectrum access in more detail.

The rest of this chapter is organized as follows. In Section 2.2, we review the literature to include background information and previous research on Cognitive Radio in general, and dynamic spectrum classification and access in specific. Under each section we give a basic description of the AI methods and learning models used in this thesis. Finally, in Section 2.3, a description of the testbed is given along with the data collection process.

2.2 Machine Learning in Cognitive Radios

2.2.1 A General Review

The huge demand to access information seamlessly and ubiquitously in the past decade in conjunction with the scarcity of radio resources [3] has resulted in intensive research towards the concept of Cognitive Radio Networks (CRNs) [10]. Nonetheless, only a small subset of published works have found their way into implementation and standardization. This is due to many factors; among them are the ambiguity of the cognitive radio concept for which many definitions exist, nonetheless a complete solution that fulfills the premise is yet to be materialized.

One of the fundamental enablers in CRN is the ability to sense the radio environment. Typically, a CR is meant to be able to sense its surrounding radio environment and learn from its past experience [9]. Supported by such awareness, a CR can bring its core functions, such as DSA, to fruition, and consequently provide a reliable and efficient utilization of the spectrum [12]. Such intelligent devices may wish to cooperate with other compatible devices, one objective being to find opportunities in underutilized spectrum bands and communicate using those opportunities. Cognitive Radios need to observe channel occupancy of a spectrum band of interest in order to perform channel selection. These intelligent radios may also want to find coexistence opportunities with other compatible devices, screen different frequency bands for white spaces, and detect occupied bands by users of a higher priority class and thus try to avoid them.

In this thesis we consider scenarios where there could be concurrent transmissions from Primary Users (PUs) owning the right of operation in a radio

spectrum band and Secondary Users (SUs) who operate on a license-exempt basis. In this regard, when a cognitive radio performs spectrum sensing it is crucial to correctly classify the priority of the transmission being sensed, i.e. whether the received signal belongs to a primary or a secondary user. If the signal is identified as a primary user's signal then transmission in that channel should be avoided. However, if the sensed signal was identified to be from a secondary transmitter then a CR should be able to contend for channel access, e.g. as in WiFi contention-based channel access using Carrier Sense Multiple Access (CSMA) method. In the absence of such a classification mechanism, mere energy-detector [31] based channel sensing may miss contention opportunities. Therefore, RAT classification should be regarded as an essential requirement in cognitive radio networking.

One way to classify radio transmissions according to their license priority is to identify their Radio Access Technology (RAT) type. Different types of technologies are acquainted with different licensing priorities. For instance, the UMTS, Groupe Spécial Mobile (GSM) and Digital Video Broadcasting–Terrestrial (DVB–T) are all radio technologies that require full control of their operating spectrum and therefore are classified as license–based primary RATs. Others, that are able to operate in the industrial, scientific and medical (ISM) radio bands such as WiFi and Bluetooth are license-exempt, where they can operate in contention-mode and tolerate interference, and therefore can resemble secondary users. Herein, we assume we can infer the priority class of the user by reasoning over the type of RAT of the sensed channel.

Various tutorials and surveys can be found in the literature on CR in general [7, 32–34], and in more detail on the use of Machine Learning [35, 36] and AI [37] techniques, accessing TVWS bands [38–40], spectrum sensing [41–43], Medium Access Control (MAC) protocols [44] and strategies [45, 46], and on security

threats [47–50].

In specific to RAT-based classification, the authors in [51] have devised an automatic RAT classifiers for the identification of WiFi and Bluetooth transmissions. Two main features were used in the classification: the maximum packet duration and the period of the silent gaps between packets. The classification performance achieves 100% correct classification when one RAT type is considered at a time. However, the study does not indicate the Signal-to-Noise Ratio of the captured packets. Moreover, the above mentioned radio communication protocols use inherently different multiplexing techniques, namely, Direct Sequence Spread Spectrum (DSSS) and Frequency Hopping Spread Spectrum (FHSS); where the former is using a fixed and wider channel bandwidth than the latter, and the latter is using pseudo-randomly hopping transmissions of very short duration, i.e. 625 μsec.

Various works have implemented classifiers that discriminate between different modulation schemes [52–54] using different machine learning algorithms. Likewise, in [55–57] the main author have put forward several deep learning architectures to classify modulation schemes consisting of BPSK, QPSK, 8PSK, 16QAM, 64QAM, BFSK, CPFSK and PAM4 for digital modulations, and WB-FM, AM-SSB, and AM-DSB for analog modulations. The reason that these works might find it hard to be part of a real-life scenario is that primary users are not restricted to the use of a specific type of modulation. In fact a primary user can jump from one modulation scheme to another in the matter of milliseconds depending on its channel state. Therefore, in our research we have opted at designing classifiers that would consider high-level RAT features, such as the spectrum mask and time-domain packet frames rather than carrier-level features such as the used modulation scheme.

Cyclostationary signal detection and classification [58] is another widely method

in the literature, where there have been various works [59–61] on the usage of this technique in general, and others [62–68] applying this method to the concept of CR. A random process is said to be wide-sense cyclostationary if its mean and autocorrelation function are periodic with some period T [69]. In this regard, since noise is rarely cyclostationary, one can detect the presence of signals at particular frequencies even if the signal power is very weak. Therefore when operating in a low SNR regime, classifiers can be designed by making use of the cyclostationary features. Moreover, since different wireless communication standards employ different specs, such as the periodicity of channel estimation and synchronization subcarriers, a number of cyclostationary features can be chosen to classify different RATs [62].

Internationally, agenda item 1.19 of World Radio-communication Conference (WRC) 2012 [70–72] has considered the development of SDR and CR. Concerning the implementation of CRNs, further work has led to Resolution 58 of the International Telecommunication Union Radio-communication sector (ITU-R), committing to continue the study of CRS. Thus, it is expected that there will be follow up work by the ITU-R on CR in the near future. This is despite globally many telecommunication bodies being anxious about the detection of the low-power transmissions of some PUs in the context of DSA, where SUs may cause harmful interference to the incumbent or licensed communication systems.

A multitude of European projects have also been funded to tackle the research challenges of cognitive radio within the EU Seventh Framework Program FP7-ICT [73]. Some of these projects are: E3 (End-to-End Efficiency – www.ict-e3.eu), CROWN (Cognitive Radio Oriented Wireless Networks – www.fp7-crown.eu), ACROPOLIS (Advanced Coexistence technologies for Radio OPtimisation in LIcensed and unlicensed Spectrum – www.ict-acropolis.eu), QUASAR

(Quantitative Assessment of Secondary Spectrum Access – www.quasarspectrum.eu), SENDORA (SEnsor Network for Dynamic and Opportunistic Radio Acess – www.sendora.eu), PHYDYAS (Physical layer for dynamic spectrum access and cognitive radio – www.ict-phydyas.org), ARAGORN (Adaptive Reconfigurable Access and Generic interfaces for Optimisation in Radio Networks – www.ict-aragorn.eu), COGEU (COgnitive radio systems for efficient sharing of TV white spaces in EUropean context – www.ict-cogeu.eu), CREW (Cognitive Radio Experimentation World – www.crew-project.eu), QoSMOS (Quality of Service and MObility driven cognitive radio Systems – www.ict-qosmos.eu), FARAMIR (Flexible and spectrum-Aware Radio Access through Measurements and modelling In cognitive Radio systems – www.ict-faramir.eu), CRS-i (Cognitive Radio Standardization-initiative—From FP7 research to global standards), OneFIT (Opportunistic Networks and Cognitive Management Systems for Efficient Application Provision in the Future Internet – www.ict-onefit.eu), and so on. Recently, however, within the EU HORIZON 2020 [74] only one project mentions cognitive radio in its description, and that is project SCREEN (Space Cognitive Radio for Electromagnetic Environment maNagement – screen.tekever.com). Rather furthering the research on cognitive radios, SCREEN attempts to focus its efforts on the "implementation and testing" of cognitive radios.

The increasingly more complex future communication networks design have also led to the advent other innovative paradigms such as Self-Organizing Networks (SONs) [75] and Docitive Networks [76]. As the communication engineering community cries out for more autonomous and intelligent operation these new communication paradigms have recently received a significant amount of attention. Traditionally, the techniques used to materialize these needs have been manually engineered to serve a particular purpose. Although,

in general most of these techniques involve ingenious ideas such as Game Theory and sophisticated digital signal processing, it was hard to achieve the level of autonomy and cognitive required. For instance, the research community had a lot of hope for the 4G Long Term Evolution (LTE) cellular mobile system, yet it did not deliver the 4G promise as established by ITU-R's IMT-Advanced requirements. This thesis, however, is chiefly inspired by the machine learning literature where concepts such as autonomy and cognition have been researched in agents and multi-agent systems.

In the following subsections below we review some of the key machine learning literature in relation to their application in cognitive communication.

2.2.2 Neural Networks

Neural nets have been used in various applications and industries in the past. In communication systems as well, and specifically as part of Cognitive Radio technology where cognition is an essential characteristic of such systems, hard-wired algorithms have limited learning capabilities. Therefore, adaptivity becomes a crucial functionality. An intelligent system must be able to withstand the dynamics of networking and gain knowledge by extracting important features from the radio environment, necessary for a better communication performance.

In this subsection, we will first start by giving a short introduction to neural networks history, review its main applications in CR, and then give a description of the analytical framework of Multi-layer Perceptron architecture.

A brief history of Neural Networks Neural Networks were first researched in 1943 by McCulloch and Pitts [77] in order to emulate the functionality of a

human brain made up of a network of artificial neurons. Hebb, in 1949, proposed the idea of reinforcing a neuron's weight based on repeated activation of its synapses [78]. In the 1950s, as computers emerged, neural networks research was pushed further forward. Rosenblatt, in 1958, laid the foundation of Perceptrons [79]. However, in 1969, Minsky and Papert showed the limitations of neural networks in their book called "Perceptrons" [80]. This pessimistic result eventually became widely generalized, which consequently hindered the research in neural networks for almost a decade. The resurgence of neural nets happened in 1986, when Rumelhart, Hinton and McClelland [81] devised the Backpropagation Algorithm for training multilayer feed-forward neural networks[1].

Use of Neural Nets in the Context of Cognitive Radio Networking
Due to its versatility MLP neural nets have been also borrowed for the research in CR. It has been used for various purposes, such as spectrum sensing [53, 54, 85, 86], spectrum behavior learning [87], dynamic spectrum access [88], and so on.

Perhaps one of the earliest works considering the use of neural nets to classify different wireless standards is Palicot et al. in [89]. This paper presents a Self-Adaptive Universal Receiver (SAUR), where the authors have studied the classification of many RATs e.g. GSM, UMTS, Digital Enhanced Cordless Telecommunications (DECT), Digital Audio Broadcasting (DAB), etc. The signal recognition was based on the *a priori* knowledge of these RATs' channel bandwidths. This approach would achieve a low classification performance since most of the recent RATs use scalable channel bandwidths, e.g. Scalable-

[1]The invention of Backpropagation algorithm reportedly [82] goes way back, where Linnainma *et al.* in 1970 [83] proposed it outside the neural network context, and Werbos *et al.* used it for neural nets in [84] in 1982.

OFDMA, and thus one RAT could be mistakenly classified for another if their bandwidth sizes matched.

Popoola et al. in [53] proposed automatic modulation classification in the context of dynamic spectrum access to detect the presence/absence of primary users. A multi-layer feed-forward neural network has been fed with manually extracted features such as normalized maximum spectral magnitude squared, standard deviation (of the absolute value) of the non-linear component of the instantaneous phase and amplitude. The study shows results of a set of very good modulation classification performances of above 99.65% when tested on signals of SNRs ranging between -5 and 20 dB. However, stress tests in extreme noisy environments where not conducted to show the limitations of this approach. In another paper [54], the same authors propose an extended version of their previous work in terms of modulation classes and manually engineered features. Although the works suggests good classification performances can be achieved, however, again, results of very low SNRs below -5 dB classification performance has not been presented.

Some of the main works of this thesis are based on the concept of neural networks from Machine Learning literature. Therefore, below we give a basic overview of Multi-layered neural nets and how they are trained to classify signals in general.

Neural Network Architecture and Training

The data for training is a set of points (vectors) x_i along with their categories t_i. For some dimension d, the $x_i \in \mathbb{R}^d$ and $t_i = [+1, -1]$. A mathematical model for the output, y, of an artificial neuron is given by Equation 2.1 and a diagram of this model is shown in Figure 2.1

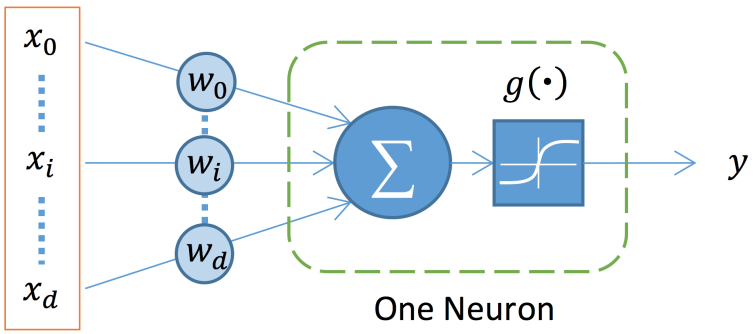


Fig. 2.1 A Neuron with Sigmoid Activation Function

$$y = g\left(\langle w, x\rangle \ + b\right), \tag{2.1}$$

Where $w \in \mathbb{R}^d$ is a weight vector, $\langle w, x\rangle$ is the inner (dot) product of the vectors w and x, b is the "bias" value of the neuron which also corresponds to w_0, and the Activation Function $g(\cdot)$ is a Tanh-Sigmoid function [90] in this case. Equation 2.1 could also be represented as

$$z_j = g\left(a_j\right), \qquad \text{for} \quad x_0 = 1, \tag{2.2}$$

where j is the index of the hidden neurons, such that $j = 1, ..., J$, where J is total number of hidden neurons, in the 2-layer perceptron neural network, and

$$a_j = \sum_i w_{ij} x_i \tag{2.3}$$

where i is the index of the input nodes.

The Tanh-Sigmoid function, $g(x)$, is expressed as

$$g\left(x\right) = \frac{2}{1 + e^{-2x}} - 1. \tag{2.4}$$

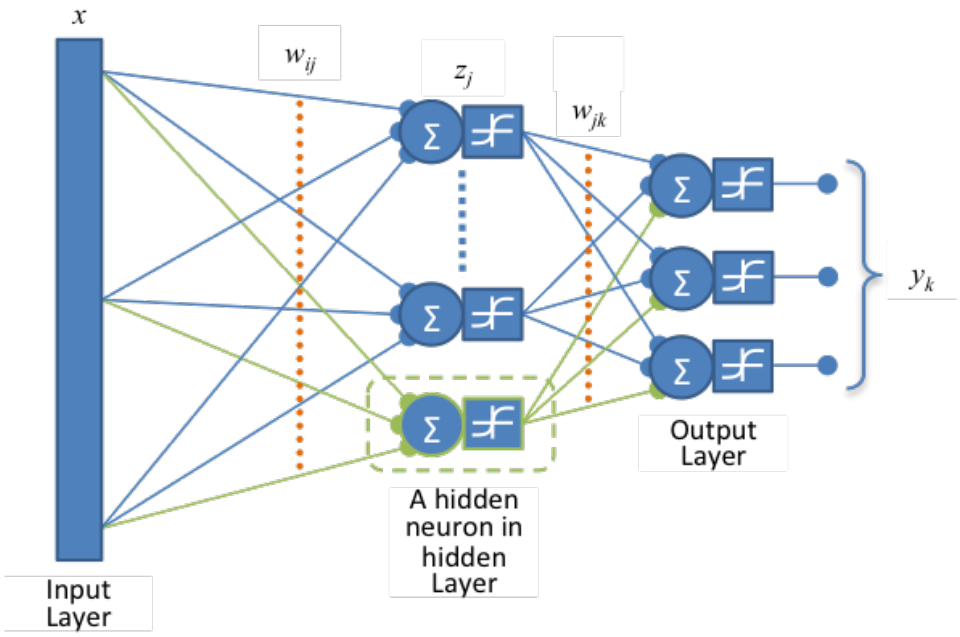


Fig. 2.2 A Multi-layer Neural Network with Sigmoid Activation Function

The Tanh-Sigmoid function is similar to $\tanh(x)$, but it differs in that it runs faster than $\tanh(x)$ as implemented in MATLAB [91, 92]. Connecting the input vector, x, to more than one neuron and then putting through the output of these neurons into yet another layer of neurons will create Multi-Layer Neuron model as shown in Figure 2.2. The final output of the multi-layered neural net, shown in Figure 2.2, can be analytically expressed as in Equation 2.5, where n is the index of the training example.

$$y_{k,n} = g\left(\sum_j w_{jk} g\left(\sum_i w_{ij}\ \mathrm{x}_{i,n}\right)\right), \qquad \text{for} \quad x_{0,n} = 1 \tag{2.5}$$

Training Neural Nets

In order to use neural networks to classify different types of inputs, we first need to train such a neural network. Consequently, in order to train a neural

network one needs a cost function, or in other words, a function that will calculate the amount of error after each classification attempt. One of the most widely used cost functions is the *Mean Squared Error* (MSE),

$$\text{MSE} = \frac{1}{2N}\sum_{n=1}^{N}\left(t_n - y_n\right)^2 \ , \tag{2.6}$$

where t_n is the correct label value and y_n is the predicted value. Putting Equations 2.4 and 2.5 into 2.6, we will get the MSE error expression,

$$\text{MSE} = \frac{1}{2N}\sum_{n=1}^{N}\left(t_n - \frac{1}{1+\exp\left(-\sum_j w_{jk}\frac{1}{1+\exp\left(-\sum_i w_{ij}\ x_{i,n}\right)}\right)}\right)^2 . \tag{2.7}$$

Simply put, a 2-layer Neural Network Training is the process of minimizing the error function shown in Equation 2.7 with respect to the weight values, in this case: w_{ij} and w_{jk}. More recently, the neural network community has adopted the *Cross-Entropy* (CE) cost function [93], which can be defined as,

$$\text{CE} = -\sum_{n=1}^{N} t_n \log(y_n) \tag{2.8}$$

or in our case, in a similar fashion to Equation 2.7, as,

$$\text{CE} = -\sum_{n=1}^{N} t_n \log\left(\frac{1}{1+\exp\left(-\sum_j w_{jk}\frac{1}{1+\exp\left(-\sum_i w_{ij}\ x_{i,n}\right)}\right)}\right). \tag{2.9}$$

When compared to MSE, the CE cost function has more attractive analytical properties [94]. CE has been proven to speed up the neural network training process through back-propagation, as well as enhancing the overall classification performance with relatively short stagnation periods [94, 95].

Algorithm 1 MLP NN Training using Backpropagation Algorithm

Require: Learning rateη
1: **for all** training examples x **do**
2: **procedure** FORWARD PROPAGATION STAGE
3: Apply an input vector x to calculate y_n using Equation 2.5
4: Calculate MSE error E Using Equation 2.7
5: **end procedure**
6: **procedure** BACKPROPAGATION STAGE
7: Calculate all δs in the hidden layer using Equation 5
8: **end procedure**
9: **procedure** WEIGHT UPDATE:(calculate new weights using Eq. 4)
10: **for all** w **do**
11: $w := w - \eta\delta z$
12: **end for**
13: **end procedure**
14: **end for**

Minimizing Equation 2.7 is not trivial, and it actually hindered the progress in neural nets research for some time. However, in 1982, Rumelhart *et al.* used the Backpropagation Algorithm [81, 83, 84] to compute the partial derivatives of at each neuron efficiently. The Backpropagation Algorithm is summarized in Algorithm 1.

Please refer to Appendix A for the derivation involved in backpropagating the error value and adjusting the weights of a neural network. Also, for a more detailed treatment of Backpropagation algorithm, the reader is kindly suggested to see Bishop's book on Pattern Recognition and Machine Learning [96].

There are two types of training methods, *Online* and *Batch*. In the Online training method, weights are updated after every input training example. While in the Batch method, training is done on all of the training samples, or a batch of them, at every training epoch. In this chapter, we opt for training the MLP neural network via stochastic online processing, although the batch method has been found [97] to yield a lower residual error E than online training. The intuition behind this is that in online training, since the weights of

the network are updated after back-propagating each training forward pass, it might be the case where a correctly learned weight vector in the previous step be destroyed by training the network on a noisy data point in the subsequent step. Yet, as will be discussed next section, due to the sequential operation nature of cognitive radios, it is imperative to classify signals as they are received. In this regard, and in order to experiment in a realistic scenario, we have trained out neural network in an online fashion.

More recently, the shift in research is toward unsupervised automatic feature extraction [98], such as Recursive Boltzmann Machines (RBMs) [99, 100], Auto-Encoder Neural Network [93] and Deep Belief Nets (DBNs) [101] which are both under the category of Deep Neural Nets (DNNs) [102]. In the next subsection we will briefly discuss Deep Learning and Deep Sparse Auto-Encoder Neural Networks.

2.2.3 Deep Learning

In this section, we will be focusing on Deep Neural Network (DNN) based methods that have been recently given lots of attention in the literature. Some of these deep learning architectures have had significant success in image recognition competitions, e.g., Convolutional Neural Nets (CNNs) in ILSVRC-2012 competition [102–104]. For instance, one of the well-known CNN architectures [105] that was used to classify the MNIST digits database [106] was consisting of convolutional layers each of which is usually followed by a sub–sampling or pooling layer, in addition to other layers, such as Rectified Linear Unit and Dropout[2] layers, and finally a fully connected conventional MLPNN layer. It is worth mentioning, however, that the idea of cascading several stages

[2]Dropout is not a unit or layer per se, however, it interacts as a layer to the neural layer in front of it i.e. its effect to its forthcoming layer is such that there is a layer of neurons that changes the number of connections to the upcoming layer.

of neurons actually have already been proposed in the eighties by Fukushima's Neocognitron architecture [107].

In general, the architecture of DNNs is different from the conventional Multi–Layer Perceptron Neural Networks (MLPNN) in a few ways [102, 105, 108]. In a DNN there are more intermediary hidden layers. These hidden layers exhibit a distributed representation through local connectivity between the preceding and intermediary layers, where each layer may have different functionality.

Recently, a few works have been noticed in the literature that use such techniques. In [109] the authors have attempted to use Deep Belief Network (DBN) for primary user signal classification. However, SNR values considered are equal to or above 0 dB, and therefore the research lacks testing the algorithm under negative SNR scenarios.

In [110] the authors have proposed what they call a Cognition-based Network (COBANET), where a linear classifier have been designed concatenating a Recursive Boltzmann's Machine (RBM) to extract content related features from video sequences. The extracted information is then meant to be used by a radio resource manager for Quality-of-Experience (QoE) improvement. In a subsequent work on this topic, the authors of [111] have put forward a Generative Deep Neural Network (GDNN) that would learn in an unsupervised fashion from various types of signals that could be captured in a communication network. Some of their initial results were presented where the trained GDNN was used to estimate the quality-rate characteristics of video flows as discussed in [110]. The latter work is a more comprehensive description of the proposed concept, but it avoids indicating that a solution to the Cognitive Radio paradigms has found, rather it eloquently points several different research challenges in the implementation of COBANETs. Inline with this research, in Chapter 5 we propose a very similar concept using Deep Sparse Auto-Encoders

and a modified version of Q-Learning to achieve DSA via exploiting channel RAT information. It is worth mentioning that the timing and topic of research of our work happens coincide with the works in [110, 111].

In a very recent and interesting work [112] (published after obtaining our results in Chapter 5), shows that better power control can be achieved between a number of distributed CRs using a DNN architecture. The DNN is here used to infer future aggregate-noise levels in a radio environment based on CR local information according to Taken's Theorem. Although this work show great potential, it is however based on the assumption that the aggregated noise is generated by the CRs and that external sporadic sources of noise does not contribute enough to make the noise estimation too complicated. Another assumption is that the considered set of CRs are all within a close range of each other for their transmissions/interference to be perceived by one another and therefore be able to correctly estimate the necessary number of dimensions of the dynamic system. However, this might not easily be the case in wireless communications when there is a Hidden Node Problem [113].

Deep Sparse Auto-Encoder Neural Networks

In the recent years there has been a surge in the research on deep architectures. Deep artificial neural networks has been one of the top ranking research topics in artificial intelligence research [102]. Auto-Encoders (AE) are a type of neural network model, that are unsupervised learning methods and that learn from a representation of data. Literature on Auto-Encoder models go back to the 80s [98] when the term was first coined, although the use of these models was somehow marginal until lately. In recent years, research in this area has picked up speed as the analytical bases of these methods were better understood [98].

Simply, an Auto-Encoder/Decoder neural network tries to match its inputs and outputs by first learning the intermediary compressed representation of the data. In light of researching distributed representation for more meaningful features, researchers have recently succeeded in training multiple layers of Auto-Encoders and stacked them on top of each other to form a deep learning architecture[3].

There are several flavors of Auto-Encoders: Sparse Coding [114], Sparse Auto-Encoders [108] and Contractive Auto-Encoders [115]. In Chapter 5 we will give further details on the technical background and cost function formulation of the designed Sparse Auto-Encoder Architecture.

2.2.4 Self-Organizing Maps

The development of SOFM Neural Nets (or simply SOMs) [116] was initially motivated by research on understanding the spatial organization of the functions of cerebral cortex region of the brain [117, 118]. Nonetheless, SOFMs have been used in a multitude of applications since its invention, e.g. in visualizing financial data [119], interpreting patterns of gene expression [120], ecological sciences [121], and so on.

A SOFM is an unsupervised learning neural network architecture, which attempts to cluster the input data independently without external help. These clusters are each comprised of identified common features. Conventionally, in Data Science, high-dimensional input vectors are reduced for various reasons, such as finding the most important dimensions, lowering computational complexity and visualization purposes. This problem, in fact is called *Manifold Learning* or *Dimensionality Reduction*

[3] *Stacked* or *Deep* Auto-Encoders and are hereafter used interchangeably.

In [122], an incremental SOM integrated with hierarchical neural network HNN (ISON-HNN) has been utilized to discriminate between authorized and unauthorized signal transmissions of SNRs ranging between -5 and 5 dB. The authors of [123] proposed the usage of a SOM in conjunction with Linear Vector Quantization (LVQ) to classify a range of different wireless communication standards, however, they did not perform stress tests in order to demonstrate the SNR at which the classification was performed. All of the scenarios described above are limited to the classification of the RATs that they are trained for and have not described a method to extend the list of considered RATs. SOMs have been also used in the context of security in [124, 125].

A basic description of the architecture and training steps of a SOFM model is provided in Section 4.3.1.

2.2.5 Support Vector Machines

Support Vector Machine (SVM) [126, 127] is a powerful supervised machine learning technique used for classification and regression purposes. In its basic form, an SVM is linear binary classifier; however, it can be extended to multiclass classifiers as well. One of the reasons of SVM's popularity is that unlike neural nets SVM does not suffer from *local minima* problem, do not suffer from the curse of dimensionality and it comes with theoretical guarantees about its performance [128].

In [129], a combination of spectral correlation analysis and SVM was used for signal classification in the context of CR. Here, a set of modulation schemes where classified based on features extracted from spectral correlation analysis of the received signals. The results show that at 0 dB SNR a classification performance of 85% and above can be achieved on a set of 6 modulation schemes.

In [130] the authors have studies RAT classification through a two-stage feature extraction and classification method. Various techniques have been used in the classification stage, such as SVM, k-Nearest Neighbor (kNN) [131], and C4.5 decision tree [132]. One of the results of this paper is that in the case of longer sensing periods SVM proves to be a better classifier that its C4.5 counterpart. However, information regarding the SNR levels at which the signals dataset was compiled has not been included.

Mathematical Formulation: Primal. Consider a data point $x_i \in \mathcal{D}, i = [1, N_{\mathcal{D}}]$ dataset that needs to differentiated into one of two classes. The separating hyper-plane equation can be described as: $\langle w, x\rangle + b = 0$, where $w \in \mathbb{R}^d$, $\langle w, x\rangle$ is the dot product of w and x, and the bias value b is real. The following problem then defines the best separating hyperplane. Find w and b that minimize $\|w\|$, such that $y_i\left(\langle w, x_i\rangle + \ b\right) \geq 1, \forall\left(x_i, y_i\right)$. The x_i on the boundary of the separating hyper-plane, that satisfy

$$y_i\left(\langle w, x_i\rangle + \ b\right) = 1, \tag{2.10}$$

are the support vectors, as shown in Figure 2.3.

The optimization problem can be formulated as minimizing $\langle w, w\rangle /2$, for mathematical convenience. This problem is a quadratic programming problem, where the optimal solution w, b enables classification of a vector z as follows:

$$class(z) \ = \ sign\left(\langle w, x\rangle \ + b\right). \tag{2.11}$$

Mathematical Formulation: Dual. In order to solve the above quadratic programming problem in a more computationally feasible manner it is advisable to use its *Dual* form. This is done by taking positive Lagrange multipliers

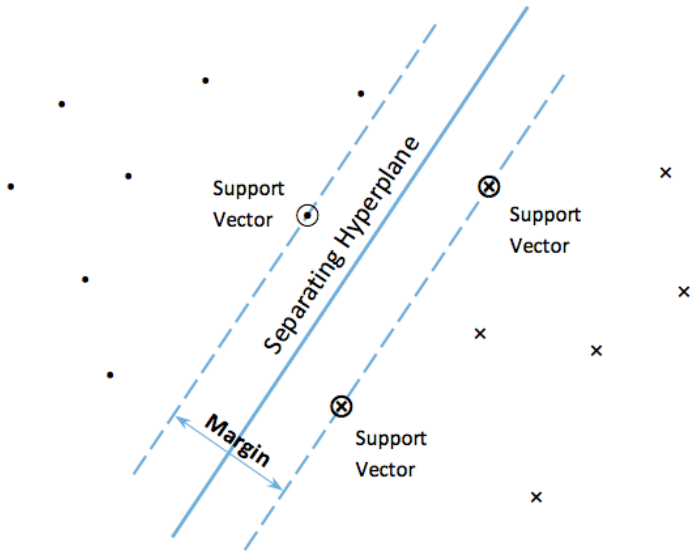


Fig. 2.3 SVM Anatomy on Linearly Separable Dataset, indicating Support Vectors, Separating Hyper-plane and the Inter-class Margin

α_i multiplied by each constraint, and subtract from the objective function:

$$L_P = \frac{1}{2}\langle w, w\rangle - \sum_i \alpha_i \left(y_i \left(\langle w, x_i\rangle + b\right) - 1\right), \tag{2.12}$$

where we try find a stationary point of L_P over w and b. Setting the partial derivative of L_P with respect to w and b to 0, we get $w = \sum_i \alpha_i y_i x_i$ and $0 = \sum_i \alpha_i y_i$. Substituting into L_P, we get the dual L_D:

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_i \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle, \tag{2.13}$$

which we maximize over $\alpha_i \geq 0$. In general, many α_i are 0 at the maximum. The nonzero α_i in the solution to the dual problem define the hyper-plane, as shown above, where w is presented as the sum of $\alpha_i y_i x_i$. The data points x_i

corresponding to nonzero α_i are the support vectors.

The derivative of L_D with respect to a nonzero α_i is 0 at an optimum. This gives, $y_i\left(\langle w, x_i\rangle + \ b\right) - 1 = 0$. In particular, this gives the value of b at the solution, by taking any i with nonzero α_i.

Non-separable Data A separating hyper-plane, in the case of non-separable data, may not be able to separate the two classes from each other. Hence, we can use a margin which shall be more flexible. In other words, we should attempt to separate most of the data and penalize the data points that lie outside this "soft margin".

There are two standard formulations of soft margins. Both involve adding slack variables s_i and a penalty parameter C.

- The L^1-norm problem is:

$$\min_{w,b,s}\left(\frac{1}{2}\langle w, w\rangle + C\sum_i s_i\ \right) \tag{2.14a}$$

  such that, $y_i\left(\langle w, x_i\rangle + \ b\right) \geq 1 - s_i$, and $s_i \geq 0$. The L^1-norm refers to using s_i as slack variables instead of their squares.

- The L^2-norm problem is:

$$\left(\frac{1}{2}\langle w, w\rangle + C\sum_i {s_i}^2\right) \tag{2.15}$$

  subject to the same constraints as above.

  Note that increasing C will place more weight on the slack variables s_i, which leads to a more strict separation between the different classes. Likewise, decreasing C towards 0 makes misclassification less important.

Mathematical Formulation of the 'Dual' for Non-separable Data case

To facilitate simpler computation, the soft-margin formulation of the L^1 dual problem is considered. Using Lagrange multipliers μ_i on the L^1-norm problem, the Lagrange function to be minimized is:

$$L_P = \frac{1}{2}\langle w, w\rangle + C\sum_i s_i - \sum_i \alpha_i \left(y_i \left(\langle w, x_i\rangle + \ b\right) - (1 - s_i)\right) - \sum_i \mu_i s_i \quad (2.16)$$

where we look for a stationary point of L_P over $w,\ b$, and positive s_i. Setting the partial derivative of L_P with respect to w, b, $\alpha,\ s$, and μ to 0, we get:

$$b = \sum_i \alpha_i y_i x_i, \qquad \sum_i \alpha_i y_i = 0, \quad \alpha_i = C - \mu_i, \qquad \alpha_i, \mu_i, s_i \geq 0. \quad (2.17)$$

The set of equations in 2.17 leads to the formulation of the dual as follows:

$$\sum_i \alpha_i - \frac{1}{2}\sum_i \sum_i \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle \quad (2.18a)$$

subject to the constraints:

$$\sum_i \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C. \quad (2.18b)$$

The gradient equation for b gives the solution b in terms of the set of nonzero α_i, which correspond to the support vectors. For a more detailed discussion, please refer to Chapter 7 of [96].

Both dual soft-margin problems are quadratic programming problems, where we are going to solve them by using MATLAB's '*svmtrain*' command [133].

2.2.6 Reinforcement Learning

Considering the highly dynamic nature of the transmissions of the primary and the secondary users in the TVWS, we would only perceive partial feedback in terms of the success rate of a communication link. In this context, we will investigate the performance on using Reinforcement Learning (RL) methods with side information to improve sensing efficiency based on prioritized sensing mechanism. Recently, RL has gained much attention due to the multitude of successful applications in various domains: such as in robotics [134–136], game playing [104, 137–139], stock trading [140–142], etc. There has been indeed many uses of RL in the cognitive radio context as well.

The use of RL is distinguishable from Supervised Learning methods, as described in [135], in that only partial feedback is given to the learner about the learner's predictions. Supervised Learning assumes that each example is a separate learning sample, while RL inherently captures the sequence of events, where an action taken may be the cause of a future event. In RL, states and actions are encoded to be able to measure the value of being at particular state and taking a specific action; while the goal is to maximize the rewarding signal that is perceived from the environment [82].

A brief analytical description of Reinforcement Learning and Q-Learning, including greedy Q-learning, functional Q-learning, and Approximate Q-learning, and Transfer learning has provided inline within Chapter 5 in sections 5.4.1, 5.4.2, and 5.5.7, respectively.

2.2.7 Matched Filter

Matched Filtering is a versatile technique that has found its way into many applications, such as image processing where 2-dimensional matched filters are used to extract features, for instance, from X-ray images [143] or retinal images [144]; in radar systems [145, 146] for time delay estimation where a reflected known signal is passed through a matched filter to detect objects at far distances; and finally, in gravitational-wave astronomy [147] to identify known shapes by matched filtering many detectors output at scale. Matched Filters are widely used for signal detection in communication systems as well [148–151]. In digital communications MFs have been chiefly utilized to detect pulses in the received noisy signal because of optimal detection and SNR maximization properties [149].

For comparison purposes, as a baseline scheme in thesis we are considering Matched Filter spectrum sensing technique to benchmark the performance of the proposed machine learning classification architectures. Below, we present a brief technical background review of this method.

To be an optimal detector, MFs require *a priori* knowledge of the actual undistorted transmitted signal, i.e. both Physical and Medium Access Control (MAC) layer such as pulse shaping, modulation type and packet format information needs to be available at the receiver [151, 152]. In cognitive radio networking, primary user signal information is hardly *fully* available at the receiving cognitive radio [151], therefore MFs are severely limited in such coherent detection scenarios. However, partial primary user transmission information can be recovered from the periodic pilot and preamble components of the received packet frame. Therefore, in our comparison work we use MFs

in a *non-coherent* fashion, i.e. we attempt to learn a MF kernel[4] including differentiating features of various received primary user signals from the past experience of the cognitive radio. This can be achieved by averaging the various received signals over a period of time, allowing for the periodic always-on pilot and preamble components to protrude from varying data carrying components of the received signal and thereby forming template for a particular RAT at a time. The template building process is discussed in Sections 3.3 and 4.4 for time and frequency domains, respectively.

Analytically, matched filtering is done as follows. Let $\mathbf{x}(n)$ be the received signal such that,

$$\mathbf{x}(n) = \begin{cases} \text{wimax(n)} & \mathbb{H}_1 \\ \text{wcdma(n)} & \mathbb{H}_2 \\ \text{wifi(n)} & \mathbb{H}_3 \end{cases} \tag{2.19}$$

where n is the bin of the received frame, and $\mathbb{H}_1$, $\mathbb{H}_2$ and $\mathbb{H}_3$ are matched filter test hypotheses, each of which is defined according a chosen threshold λ, such that,

$$\begin{cases} \mathbb{H}^+ & \text{if} \quad \sum_{n=1}^{N} \mathbf{x}(n) K_{\mathbb{H}}(n)^* \geq \lambda \\ \mathbb{H}^- & \text{if} \quad \sum_{n=1}^{N} \mathbf{x}(n) K_{\mathbb{H}}(n)^* < \lambda \end{cases} \tag{2.20}$$

where $\mathbb{H}^+$ and $\mathbb{H}^-$ indicate whether the hypothesis is tested true or false, respectively; $K_{\mathbb{H}}(n)$ is the trained kernel for a particular RAT class match filter $\mathbb{H}$, N is the total number of bins per frame and '*' is the complex conjugate operator. The filtering process done in Equation 2.20 is equivalent to convolution of the received unknown frame with the conjugated time-reversed version of the kernel.

[4]In regards to Matched Filters, the terms 'kernel' and 'template' are used interchangeably in this thesis.

The major advantage of the MF is that it can achieve high processing gain due to coherent/semi-coherent detection. However, one needs to design a separate MF for each class of RAT needed to be identified, which would increase the complexity of the overall design. In addition, the threshold λ should also be manually tuned in order to achieve optimum results.

2.2.8 Perspective research trends

In the preceding subsections we reviewed research works done in the *past*. In this subsection, we briefly review possible *future* trends and challenges in next generation communication networks.

Each xG communication technology has aimed at improving the broadband experience. Meanwhile, rapid innovation in telecommunications has also enabled us to flood ourselves with data of all types, sizes, and speeds—to the extent that our conventional processing paradigms break down when mining these massive datasets in order to drive profitability.

In a joint study, Ericsson and Chalmers University have shown that doubling the broadband speed for an economy increases its Gross Domestic Product (GDP) by 0.3%, which is equivalent to USD 126 Billion in the OECD region [153]. Moreover, the report indicates that for a 10% increase in Internet penetration, the GDP improves by 1%. It is thus obvious that the incentives are in place to further advocate for higher broadband speeds and wider availability. This has become the driver for innovation in terms of R&D Projects to come up with new technologies. These new technologies have in turn led industry to develop a plethora of various applications, all relying on the premise of ubiquitous high–speed Internet access. The applications that made it to mass markets and were desirable became more demanding in terms of performance

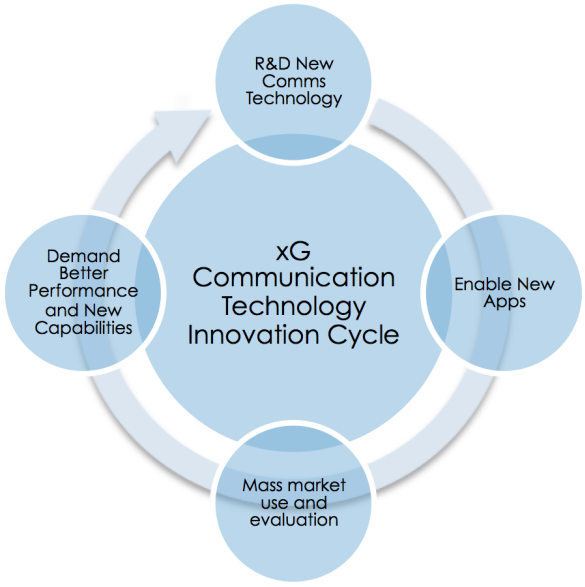


Fig. 2.4 xG Communication Technology Innovation Cycle

and services they provisioned. Finally, when the supply provided by the technology of the time was not enough to meet the demands, the R&D community came up with new techniques, hence closing the innovation cycle as shown in Figure 2.4.

From the standpoint of the mobile communications industry, these performance demands have essentially become the bases for 5G communication technologies' ambitious requirements: a 1000x increase in data rates, under 1ms E2E network delay, 90% reduction in network energy use, and so on as discussed in [15, 16, 30].

Taking the above discussion into account we have recently realized that there is a twist to the capacity crunch problem. Today, we have unintentionally dug ourselves into a hole as we are witnessing a tsunami of data that we have never experienced before. Neither our computing power nor processing methodology is adequate to tackle the heaps of information at our datastores. The issues caused by this data explosion can be analyzed into three degrees of freedom:

Volume, Velocity and Variety; known as the 3Vs of Big Data:

- Volume – the scale of stored data elements is too large for traditional datastores to handle. For instance, a 10 Terabyte dataset cannot fit into most of the current available hard drives in the market; actually finding a PC with more than a few Terabytes of disk space out of the box is very rare. Data files need to be distributed but also easy to read and write to.

- Velocity – new (streaming) data is arriving at a rate that cannot be processed at real–time. Algorithms need to be parallelized such that each computing unit can take on a portion of the computation, outputting timely answers to real–life events. In Figure 2.5a, Cisco's predictions for global mobile data growth [1] indicate a 10 fold exponential increase by 2019, reaching a 24.3 Exabytes (million Terabytes) per month.

- Variety – data is gathered by combining multiple different sources with different data structures. This is an obstacle when importing data to conventional Relational Databases, since each data source needs to be manually restructured in order to be consistent with data imported from other sources. Figure 2.5b shows the outcome of a survey conducted by IBM and Oxford University in 2013 indicating the percentage usage of different big data sources by a group of companies [2].

As shown in Figure 2.6, a Big Data problem is any instance of a system possessing one or more of the above 3 Vs. A Big Data enabled network thus would be a solution that tackles any of the 3V bottlenecks of that network. The ICT industry, in general, is the founder of Big Data. Telco's have been dealing with Big Data day in and day out. However, as the buzzword "Big Data" has gained more recognition, many other sectors have started to look at

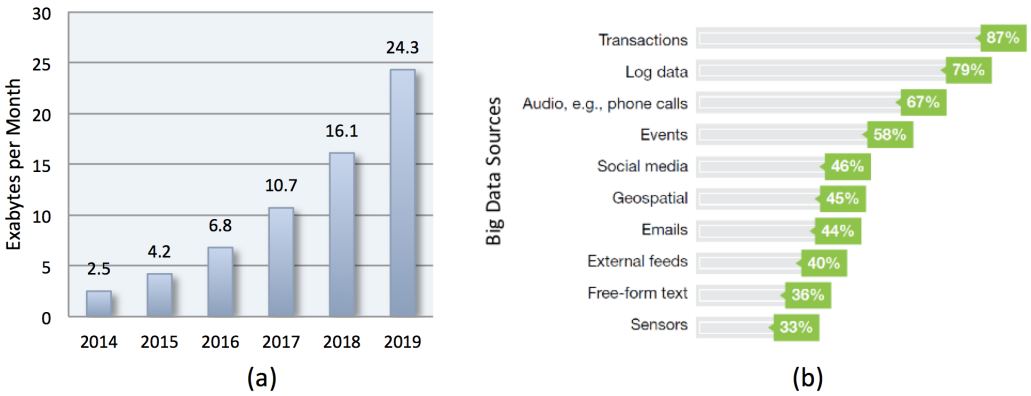


Fig. 2.5 Data Traffic Trends, (a) 10-fold increase in mobile data traffic by 2019 from Cisco [1], (b) Big Data Sources from IBM [2]

the potential of their untapped data under Big Data techniques. As of 2015, 95% of the Fortune 500 businesses have initiated at least one Big Data project within their premises [2].

Therefore, for the past few decades and throughout the generations of communication technology that has been developed, the wireless communications community have consistently aimed at increasing capacity and bit rate. This has, in effect, led to the accommodation a plethora of applications, which has resulted in the data explosion experienced today. Communication service providers are under a lot of pressure to reduce costs, protect the margins, and retain their customer base. One of the best ways to do that is to use the data chaos to their advantage by mining the data that traverses their networks to

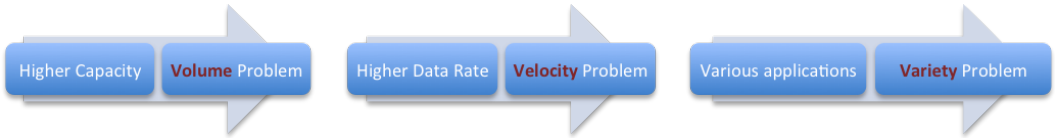


Fig. 2.6 Next Generation Communication Systems' Big Data Challenges

drive profitability. The communication engineering community has recently started researching these aspects [30, 154–157].

2.3 Testbed Setup and Data Collection

For CRs to process information, they need to maintain a cognitive engine that is able to analyze the sensed raw data. Beneficially from this, a CR is required to be flexible in order to adapt to changed circumstances and integrate new knowledge into its knowledge base on the go. Since almost all of the radio reconfigurability in today's communication devices is inherent to their software defined nature, it is often assumed that the implementation of Cognitive Radio Systems (CRS) will be based on Software Defined Radios (SDRs) [9].

In order to examine the performance of the RAT classifiers in a realistic scenario than mere PC-based simulations, a practical testbed is assembled. All of the experiments conducted for the purpose of this thesis where based on the initial testbed setting shown in Figure 2.7. This setup was a collaborative work done with Ss. Cyril and Methodius University in Skopje, Macedonia, within the ICT EU project "Advanced coexistence technologies for radio optimization in licensed and unlicensed spectrum" (ACROPOLIS) [158].

As shown in the figure above we used Universal Software Radio Peripheral (USRP) model N210 from Ettus Research (www.ettus.com/product/details/UN210-KIT) as the SDR communication device. Three different radio access technologies were taken into account: DVB-T to exemplify Digital TV signals, WCDMA to represent a primary mobile network signal, and IEEE 802.11a WiFi to represent an unlicensed secondary transmission. These 3 RAT types were simulated and tested in MATLAB first, then ported to the USRP to be transmitted over-the-air.

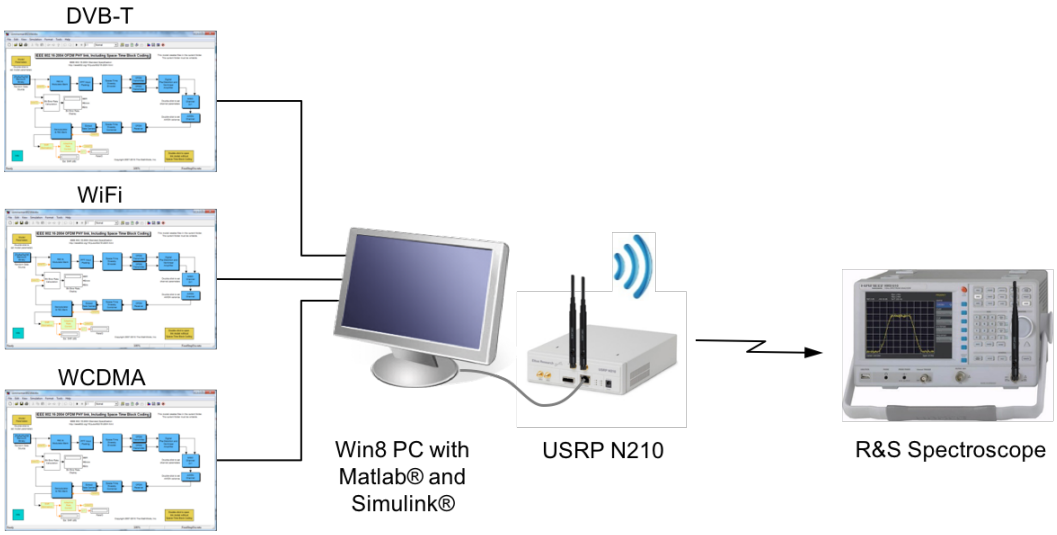


Fig. 2.7 DVB-T, WiFi, UMTS (WCDMA) Radio Access Technology Emulation and Transmission Using MATLAB, SIMULINK and a USRP device.

In the beginning we programmed the USRP to transmit different RAT signals designed in MATLAB and SIMULINK software packages and verified the output RAT waveform via the R&S Spectroscope. The Spectrum Analyzer was later replaced by another USRP acting as the receiver as shown in Figure 2.8. So the actual testbed used for RAT signal generation and reception was composed of two N210 USRPs driven by two Windows 8 PCs connected via a Gigabit-Ethernet Local Area Network (LAN). All experiments were conducted at 2.45 GHz Industrial, Scientific and Medical (ISM) band. VERT2450 antennas were used along with USRP SBX daughterboards, which cover a variety of bands in the 400 MHz–4400 MHz range.

Finally, the dataset was compiled in 3 batches such that each RAT configuration would be loaded to the transmitting USRP at a time and the receiving USRP would record the received signals. Time series data was collected from

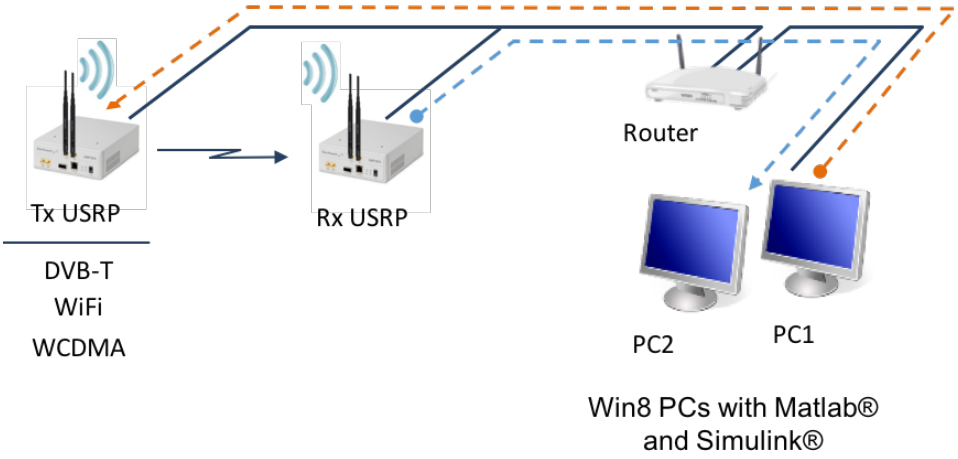


Fig. 2.8 A Schematic of the Testbed Network

the receiving USRP for the duration of *one second*—which is to be used later for offline analyses. As discussed in the coming chapters, algorithms proposed in this thesis ran on the stored data, in a Hardware-in-the-Loop (HIL) fashion.

In Chapter 3 the time-domain version of the captured data will be used. While in Chapter 4 we will use Welch's method to transform the received signals to frequency-domain. The rationale behind the domain change is based on the fact that communication transmissions are encoded in frequency-domain in order to lower frequency-selective fading [150]. For the same reason, some of the prominent and high-power frame components are assigned in frequency domain, such as pilot subcarriers [113]. These pilot signals, as will be shown later, will be important features in correctly classifying the different radio access technologies considered in this thesis.

Chapter 3

Radio Access Technology Classification for Cognitive Radio Networks using Multi-Layer Perceptron Neural Networks

3.1 Introduction

In the previous chapter we presented a general overview of the main challenges of wireless communication systems in the current day. One of the prominent challenges was the capacity problem, where future communication systems need to accommodate the exploding data demand. Therein we discussed the proposal of the Cognitive Radio paradigm and Dynamic Spectrum Access. In this chapter, we proceed with the RAT recognition research which is an

essential function of cognitive radios to achieve DSA and therefore we investigate means to achieve this goal. Here we aim to demonstrate the suitability of learning algorithms for classification of various RATs, especially for the foreseen case of primary-secondary cooperation in shared spectrum bands—a highly regarded scenario in the ITU's World Radio Conference (WRC) 2012.

Building upon the background information given in Chapter 2, in the next section we start by setting the stage for our neural network design and elaborate on the pre-processing stage to put the data in the most appropriate form. Later, we propose a Multi-Layer Perceptron neural network, and train it with Backpropagation algorithm to classify different emulated wireless signals under various levels of noise. In Section 3.4 we unveil the results of the neural signal processing, and later discuss potential applications in Section 3.5. Section 3.6 summarizes results and concludes the work.

3.2 Supervised Learning Neural Network For RAT Classification

The neural network considered in this chapter is structured as a Multi Layer Perceptron (MLP) network and it comprises of two layers: a hidden layer and an output layer. Through trial and error experimentation, the number of nodes in the hidden layer was set to 20. DVB-T, WCDMA and WiFi signals are fed into the network separately. The traffic generator is set to operate on a full-buffer basis. The received packets from the receiving USRP are cut into chunks of 22200 samples representing the features that are input to the neural network.

The 22200 time-domain sample window length was chosen such that one full

RF packet frame of any of the different RATs could be fully captured. In this regard, we have found out that the DVB–T signal transmissions to be the least frequent of all the three RATs considered, with a full frame length of 22200 samples per frame, which was calculated by taking the autocorrelation of a sample of each RAT and then choosing index of the first peak. The autocorrelation function can be described as,

$$R_h = \frac{C_h}{C_0} \tag{3.1}$$

where $C_h$is the Autocovariance function,

$$C_h = \frac{\sum_{i=1}^{N-k}(Y_i - \bar{Y})(Y_{i+k} - \bar{Y})}{\sum_{i=1}^{N}(Y_i - \bar{Y})^2}, \tag{3.2}$$

and C_0 is the variance of the time-series sample,

$$C_0 = \frac{\sum_{t=1}^{N}(Y_t - \bar{Y})^2}{N}, \tag{3.3}$$

where Y is the time-domain signal of N samples, $\bar{Y}$ is the sample mean value.

Note that the value of the chosen window length depends on the sampling frequency of the device used to capture the signal and not only the original periodic time duration of the RAT frames sent. However, as long as this figure is set constant throughout the experiment such that it captures at least one frame of each of the three RATs the value of this figure should not be an obstacle in the neural network training process. On the other hand, if the sample window is taken to be less than one full RAT transmission frame, all of the features of that RAT frame may not be captured, leading to the loss of features and thus lower classification performance.

The raw captured signals from the USRP is in the complex form,

$$z(n) = a(n) + i\ b(n). \tag{3.4}$$

These are converted to Complex Magnitude (or Modulus) Power, such that,

$$s(n) = |z(n)|^2 = a^2(n) + b^2(n). \tag{3.5}$$

The signal power $s(n)$ is also passed through a normalization stage such that its values range from 0 to 1. This is an essential pre-processing step in getting the data into the correct form for the MLP neural net to be unbiased of the received power strength of the various RATs considered. Otherwise, if the power of one of the RATs is on average higher than the others, the neural network would rather classify that particular RAT based on based on the power content of the signal rather than its features. Therefore if such neural network is fed with a lower power frame of the mentioned RAT it would not be able to classify the frame correctly. This is a well-documented and often-cited side-effect of neural network [159, 160]. In this regard, a decent classification performance could be achieved if the received power levels were held relatively constant throughout all the training examples. However, in wireless communications it is evidently known that this not the case, where transmitters and receivers can be at arbitrary different range of distances and thus at different receive power levels. Therefore, it is crucial to standardize each received RAT frame before feeding it to the MLP neural network training stage. In this regard, we use the z-score standardization method, where every received frame $s(n)$ is first subtracted by mean value of the frame and then divided by its standard

deviation, as described below in Equation 3.6,

$$\begin{aligned} s(n) &= \frac{s(n) - \bar{s}(n)}{\mathrm{sd}(s(n))} \\ &= \frac{s(n) - \frac{\sum_{n=1}^{N} s(n)}{N}}{\sqrt{\frac{\sum_{n=1}^{N} \left(s(n) - \frac{\sum_{n=1}^{N} s(n)}{N} \right)^2}{N}}} \end{aligned} \tag{3.6}$$

Time-domain Frame Synchronization

Another important aspect in time-domain pattern recognition using MLP neural nets is the synchronization of the incoming frames. If the features present in the captured frames are shifted (in terms of their location with respect to the frame bins), then it would be very difficult for the MLP architecture to have a good classification performance. This is due to such shallow neural network's inability to provide local invariance, a highly sought after feature in pattern recognition tasks [108]. In other words, the performance of shallow MLP networks vary if some or all of the features of the input signal are not confined to certain bin positions.

In our case, where we capture various RATs with different frame periodicity the above discussed issue poses a challenge in efficiently training the neural network. Take for instance the case of fixing the capture time window according to the DVB-T signals as discussed above; Since the WCDMA and the WiFi signal possess different frame periods, each of their incoming consequent frames suffer a shift in time as compared to the original frame.

Luckily, the synchronization pilot and preamble features found in these RATs allow us to synchronize these frames such that their feature are aligned from one frame to another. In this work we have achieved frame synchronization by

simply measuring the time-delay between the kernel and each of the incoming 22200 time sample bins.

Using cross-correlation, if the signal was found to be lagging the kernel then a leading trail of zeros equal to the number of delayed bins are prepended and the same of number bins are truncated from end of the frame. On the other hand, if the signal was found to be leading the kernel then the leading number of bins equal to the delay are truncated and a trail of zeros of the same number of bins are appended to the end of the frame. This process is repeated for all of the captured frames of each radio access technology mentioned.

3.2.1 MLP Neural Net Design

750 signal examples each of which comprises of 22200 bins/elements (vector dimensions) for the three classes were parsed together to form the design matrix (input data). A schematic depicting the 2-layered neural network is shown in Figure 3.1.

The choice of the number of neurons at the hidden layer is an essential part of the MLP architecture design. We have ran 10 simulations for 3 different MLP architectures with 5, 20 and 35 hidden layer neurons at a time, where their

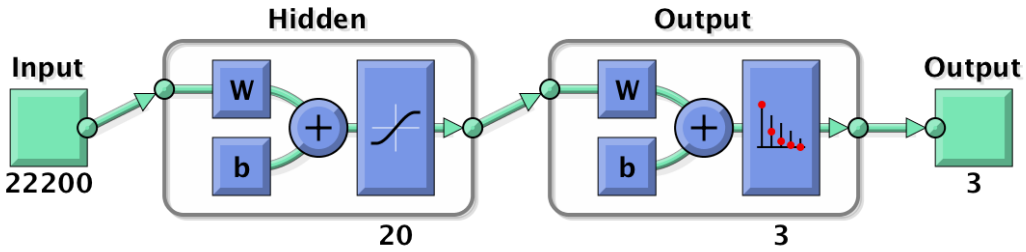


Fig. 3.1 A 2-Layer Neural Network with 20 hidden nodes

overall classification performance is shown in Figure 3.2. Here we can see that the performance of the architecture with 35 hidden neurons is slightly superior to that with the 20 hidden neurons and clearly superior than that with the 5 hidden neurons. Since the slight difference between architectures with the 35 and 20 hidden neurons is where the classification is below even below the 50% mark, this performance difference does not have a major effect. On the contrary, choosing the least complex architecture of the mentioned approximately equally performing architectures would make the final architecture computationally more efficient. Therefore, in this chapter we will stick to the MLP architecture that has 20 neurons at its hidden layer.

The hidden-layer transfer function takes the total weighted input and bias values and passes them through a Hyperbolic Tangent Sigmoid (Tanh-Sig) function, provided by Equation 2.4, and the output layer passes its weighted input and bias values through a Cross-Entropy function CE, as described in Equation 2.8.

The weights are initialized according to Nguyen-Widrow layer initialization

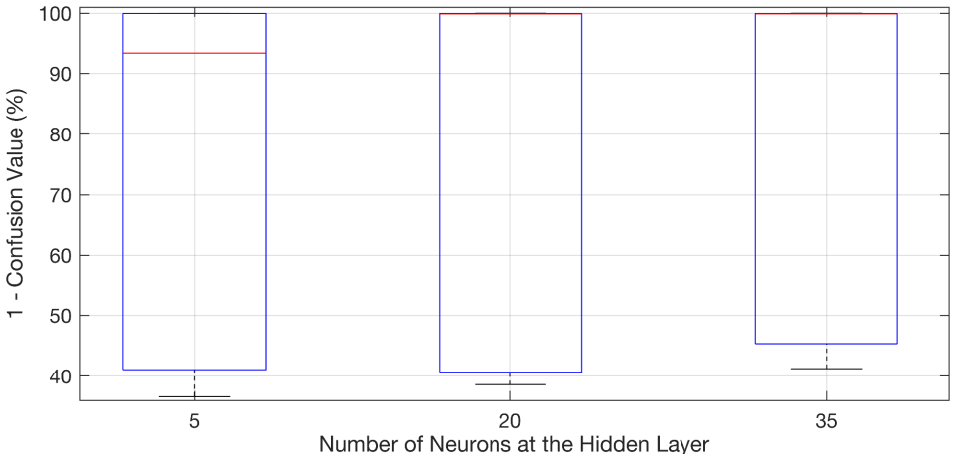


Fig. 3.2 2-Layer MLP Neural Network Performance Evaluation While Varying the Number of the Hidden Nodes

method [92]. Back propagation is used to calculate derivatives of the performance function, or in other words the cost function, with respect to the weight and bias variables. The Scaled conjugate gradient algorithm [161] implemented in MATLAB was used to apply the conjugate gradient updates. This algorithm can train any network as long as its weight, net input, and transfer functions have derivatives [92].

The scaled conjugate gradient algorithm is based on conjugate directions but this algorithm, unlike other versions of the conjugate gradient back propagation algorithm, does not perform a line search at each iteration to find the learning rate $\eta(n)$ that minimizes the cost function [92]. Instead, as stated in [162] $\eta(n)$is given by

$$\eta(n) = 2\left(\eta(n) - P^T H(\delta^n)P + \eta(n)\frac{\| P \|^2}{\| P \|}\right)^2, \tag{3.7}$$

where n is the step index, H is the Hessian of the gradient,

$$H(\delta^n) = \begin{bmatrix} \frac{\partial^2\delta^n}{\partial x_1\partial x_1} & \frac{\partial^2\delta^n}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2\delta^n}{\partial x_1\partial x_N} \\ \frac{\partial^2\delta^n}{\partial x_2\partial x_1} & \frac{\partial^2\delta^n}{\partial x_2\partial x_2} & \cdots & \frac{\partial^2\delta^n}{\partial x_2\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2\delta^n}{\partial x_N\partial x_1} & \frac{\partial^2\delta^n}{\partial x_N\partial x_2} & \cdots & \frac{\partial^2\delta^n}{\partial x_N\partial x_N} \end{bmatrix} \tag{3.8}$$

and P is given by

$$P = \frac{\partial E(n)}{\partial w}. \tag{3.9}$$

$E(n)$ is the Minimum Square Error function, and w is the neuron weight vector. For a more detailed observation of the scaled conjugate gradient back propagation algorithm the reader is kindly referred to [162].

Practical Guidelines

For many of the optimization algorithms used for network training, such as conjugate gradients, the error is a non-increasing function of the iteration index. Nevertheless, the error generated by test dataset, which is not used in the training process, decreases in the beginning, and then starts to increase when the neural net starts over-fitting the data. In this regard, the data is divided randomly into three datasets: 70% for training, 15% for testing and 15% for validation. Therefore, in order to avoid the neural network to over-fit the training data, alongside the training data, a validation dataset is used as a test dataset so that it validates the performance of the training as it proceeds. Training will be discontinued at the point of smallest error after it has not decreased for a certain number of iterations (called Maximum validation failures) with respect to the validation dataset. This is done in order to obtain a neural network having a good generalization performance and avoid over-fitting the training dataset. This method is called Early Stopping approach [96].

There are other practical guidelines one could take to reach a better performance. In our case of RAT Classification, training stops when any of the following conditions occurs [162]:

- Validation error has been equal or increased more than a specific maximum number of failing times since the last time it decreased.
- The performance gradient falls below the minimum specified gradient

Table 3.1 Neural Network Parameters

Maximum validation failures	6
Maximum Epochs	1000
Maximum Training Time	Inf
Performance Goal	0
Minimum Gradient	10^{-6}
Transfer Function	Tanh–Sigmoid
Network Cost Function	Cross–Entropy
Network Training Function	Scaled Conjugate Gradient

value. The above discussed neural network parameters and their corresponding values are indicated in Table 3.1.

- The maximum number of epochs (repetitions) is reached before reaching some acceptable level of classification error.
- The maximum amount of time set to reach the performance levels is exceeded.
- Performance is minimized to the goal, which is usually set to zero.

3.3 Matched Filter Design

In order to benchmark the performance of the proposed MLP neural network architecture mentioned above, we have implemented a Matched Filter spectrum sensing method to serve as a baseline for comparison. As discussed in Section 2.2.7, Matched Filters are widely used in pattern recognition task in a multitude of varying applications and industries [143–149, 163]. In this chapter, we are going to use matched filters with time-domain transmission frame

kernels for each RAT considered. In order to optimize the performance of the matched filters we also need to find the optimum decision threshold value λ as mentioned in Equation 2.20.

First, regarding building the Kernels, since initially we do not have them at the receiving cognitive radio, we need to acquire them in a non-coherent fashion on the fly. This can be done by first collecting a number of high SNR frames for each RAT and later averaging them out. This way the consistent features, such as the protruding pilot and preamble frame components, throughout the captured set of frames of a particular RAT will be kept and the rest of the fluctuating data and noise part will be relatively diminished due to the averaging stage. As shown in Figure 3.3, the kernels built for the purpose of the research in this chapter are based on signals captures at a SNR of 25 dB and 250 frames for each RAT.

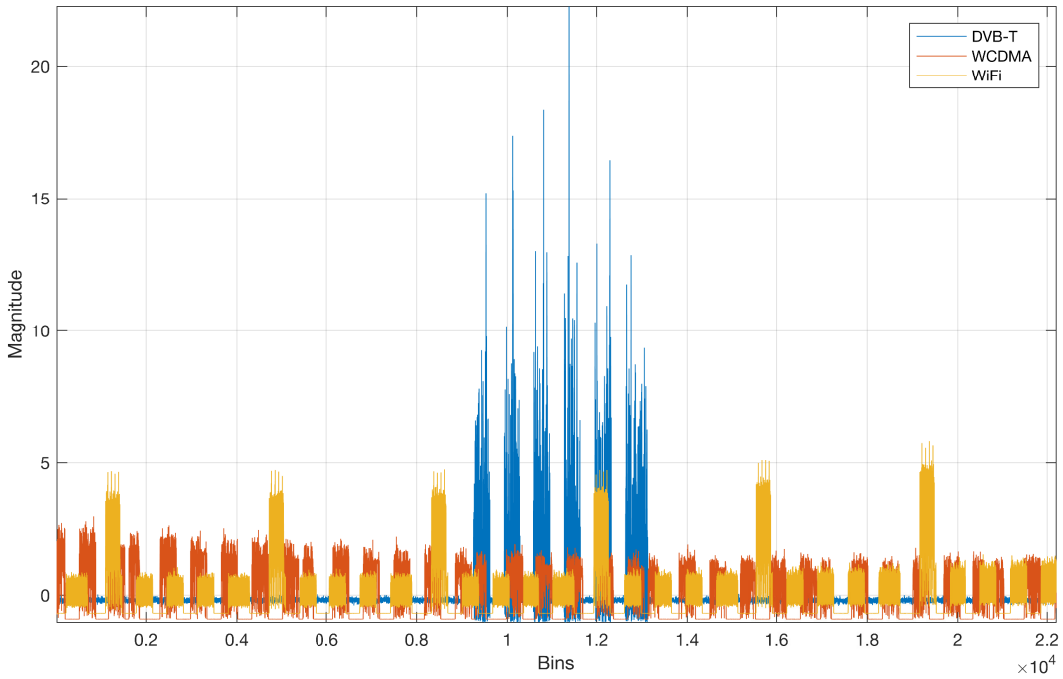


Fig. 3.3 Time-domain Kernels for DVB-T, WCDMA and IEEE 802.11a WiFi Radio Access Technologies

Finally, regarding the decision threshold λ optimization, we have ran a Monte Carlo experiment to find the λ value that maximizes the overall classification performance. According to Figure 3.4, which presents the classification performance of a range of λ values with respect to SNR values ranging from -40 dB to 20 dB, where $\lambda = 0.2$ is found to have the best performance.

3.4 Simulation Results and Discussion

Having discussed design parameters and settings for both the MLP neural net and Matched Filter classifiers in the previous section, next we elaborate on their classification performances. We first start by taking a more detailed look at the 0 dB SNR case. Later, we extend the range of the simulated SNR cases to test how the classifiers resist to extremely noisy scenarios.

Figure 3.5 presents the training, testing and validation results of the MLP

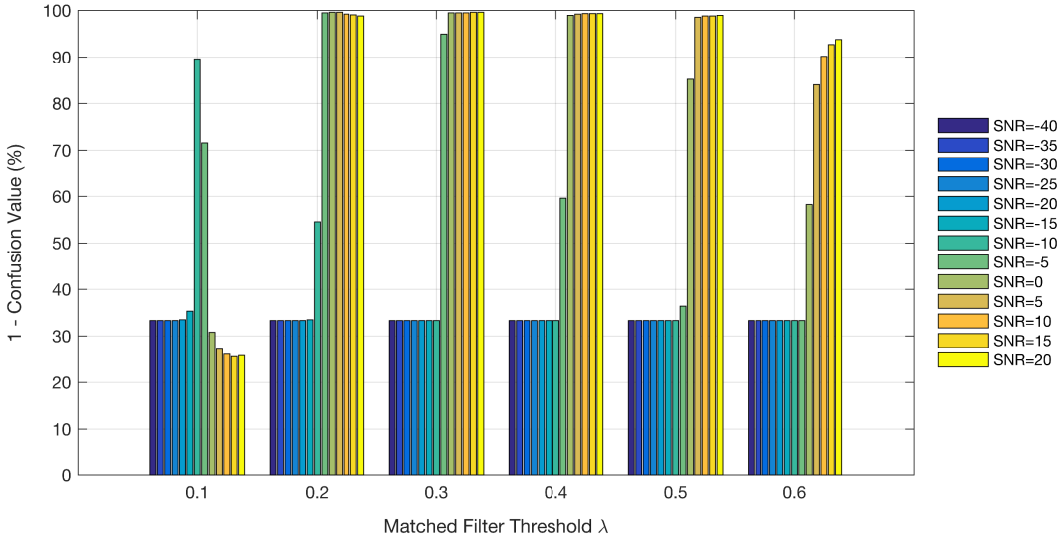


Fig. 3.4 Classification Performance of a Range of Matched Filter Threshold Values λ

neural net. The training, testing and validation cross-entropy errors decrease until epoch 25 where the Early Stopping algorithm comes to play. Otherwise, if the difference between the training and validation errors increases the network is said to be *over-fitting* the input data.

The confusion matrices shown in Figure 3.6 demonstrate the potential of the MLP neural net classification method. Confusion matrix are typically used in machine learning classification problems to indicate the performance of a supervised learning classification algorithm where the number of correct guesses of the neural network is shown along their diagonals, and the misclassifications are displayed outside the diagonal with respect to each class accordingly. Here class 1 represent the DVB-T, class 2 WCDMA and class 3 the WiFi radio access technologies.

Regarding performance neural network, as indicated by the confusion matrices, to avoid misperception it is necessary to mention the close proximity of the transmitter and receiver USRPs, ranging between 2 to 3 meters of horizontal distance from each other. This has certainly led to a strong signal reception at

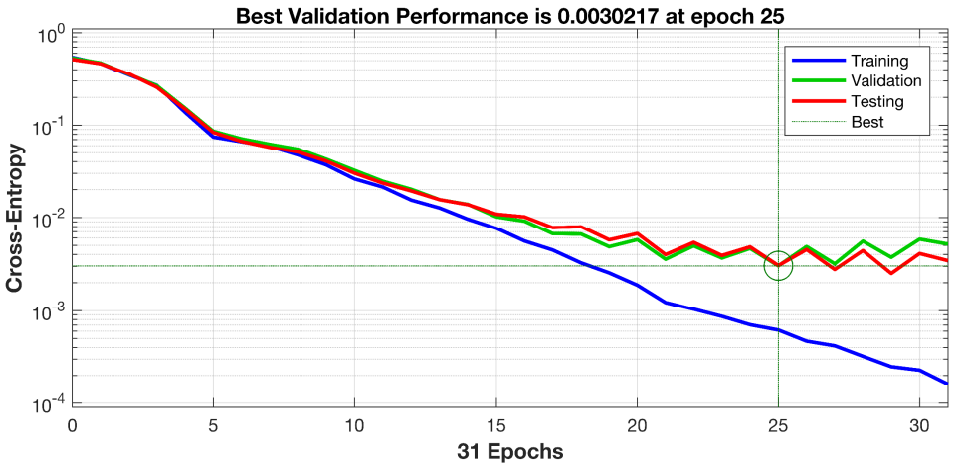


Fig. 3.5 Training, testing and validation Cross-Entropy graphs at SNR = 0 dB

the receiver and henceforth a better classification. The SNR of the originally received signals were about 20 to 25 dB. Factoring in AWGN in the scenario considered in Figure 3.6, the SNR is lowered to 0 dB. However, the MLP still achieves a descent performance at the mentioned SNR.

The lower section of Figure 3.7 displays the errors made in the validation set versus epoch index. At epoch 25, where the best validation performance is reached, which is approximately equal to 0.01, the neural network fails to

Fig. 3.6 Training, Testing and Validation Confusion Matrices at SNR = 0 dB

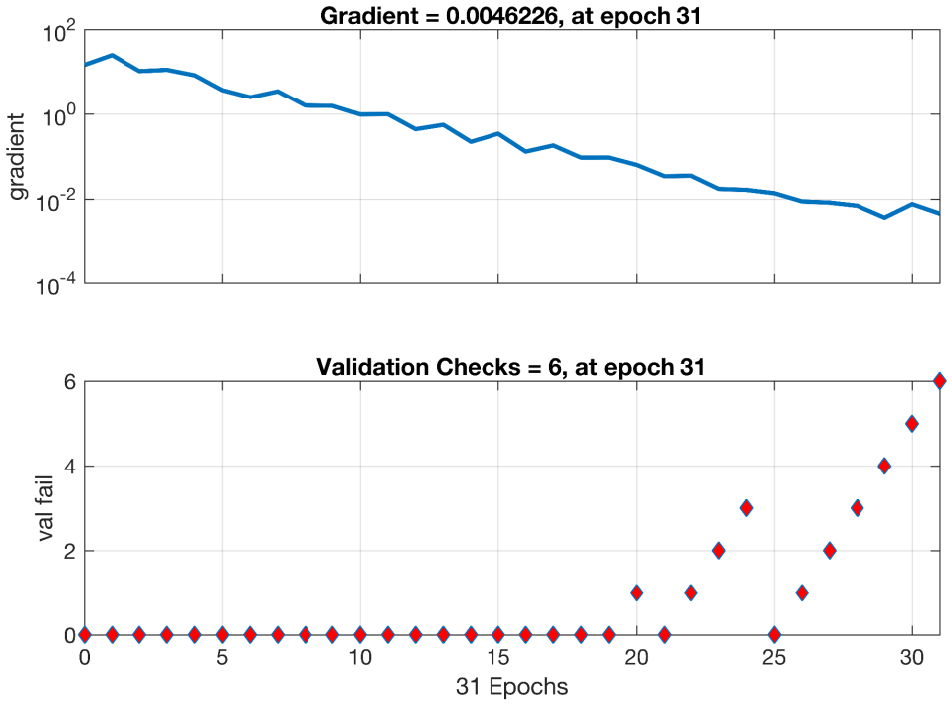
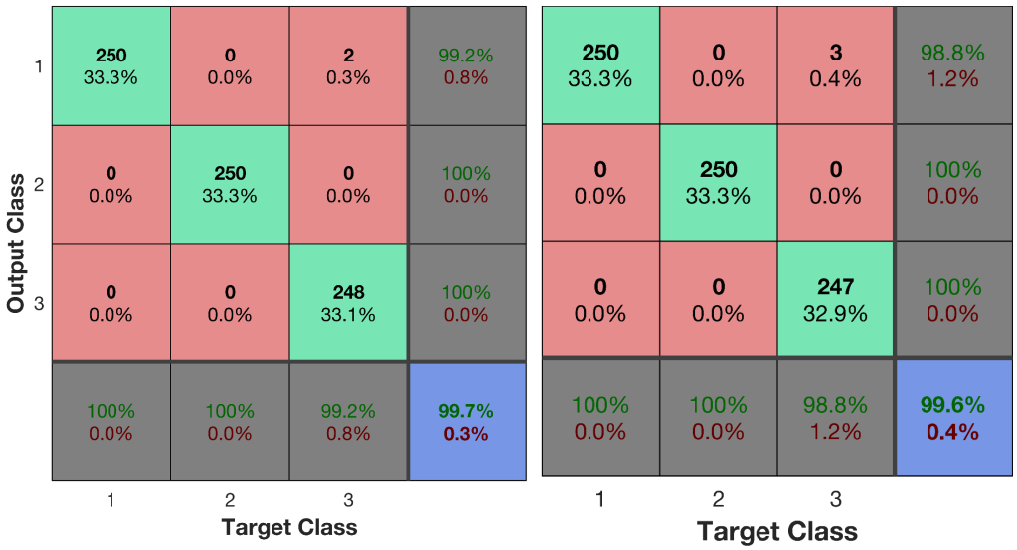


Fig. 3.7 (upper) Gradient at each epoch, (lower) validation error count, at SNR = 0 dB

optimize its performance for the next 6 consecutive epochs and thus comes to a halt as it was programmed to do so according to Early Stopping method.

Comparison with Matched Filter Classifier

Figures 3.5, 3.6 and 3.7 only show the results of one training experiment. In order to avoid an optimistic and biased result, it is advisable to take the average of a set of simulations by performing many training experiments and averaging over the number of experiments. In this regard, the designed neural net was trained 100 times and the resulting average confusion matrix generated is shown in Figure 3.8a. As compared to the Matched Filter classifier performance shown in Figure 3.8b, at 0 dB SNR, we cannot notice a significant classification error difference.



(a) MLP Neural Net **(b)** Matched Filter

Fig. 3.8 Average MLP Neural Net and Matched Filter Confusion Matrices at SNR = 0 dB

Extending the Range of SNRs

All of the above classification results were achieved at the 0 dB SNR scenario. In the Figures presented in this section, we show the performance of MLP architecture with and without data frame synchronization and compare them against the baseline Matched Filter method across a set of SNRs ranging from -40 dB to 20 dB in steps of 5 dB.

Figure 3.9 presents various configurations of MF classification performance. At the threshold value $\lambda = 0.2$ we get a very high performance for SNRs of 20 dB and down to -5 dB, but achieving poor results from -5 dB and lower. Although at $\lambda = 0.1$ we can observe a much higher performance when the SNR is equal to -10 dB, however the overall performance of this configuration is rather low and inconsistent. When the threshold is kept very low, the MF passes all of the incoming signals which leads a higher probability of false alarm and therefore

higher confusion value.

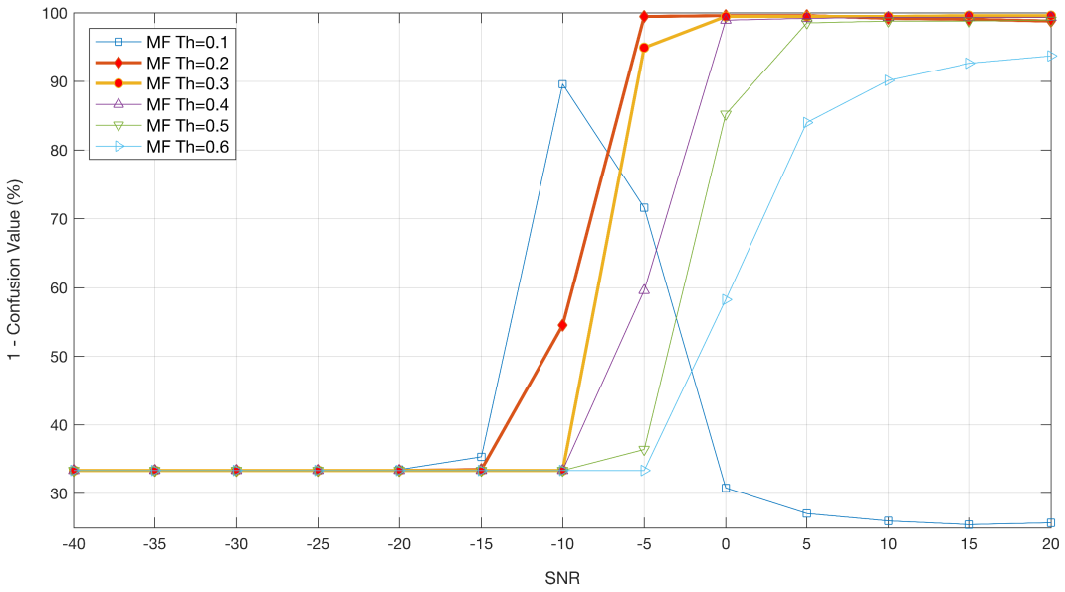


Fig. 3.9 Overall Classification Performance of Matched Filters with DVB-T, WCDMA and WiFi Kernels at Threshold Values=[0.1,0.6]

Using the synchronized dataset as was discussed in Section 3.2, a set of simulations for the same range of SNRs as above have been run and mean classification results achieved are given by Figure 3.10.

Finally, Figure 3.11 presents a comparison between the 3 classifiers considered in this chapter. We can clearly observe that the MLP neural net with 20 hidden neurons which was trained on the synchronized dataset consistently achieves better classification results than the other two classifiers. As for the MLP architecture which was not trained on the synchronized dataset, it could not beat the baseline MF classifier when the SNR was higher than -10 dB. As was mentioned before in Section 3.2, in this case the MLP neural net suffers from inconsistent appearance of the RAT features at different locations for each training example.

In our experiments the superior classification performance of the MLP design

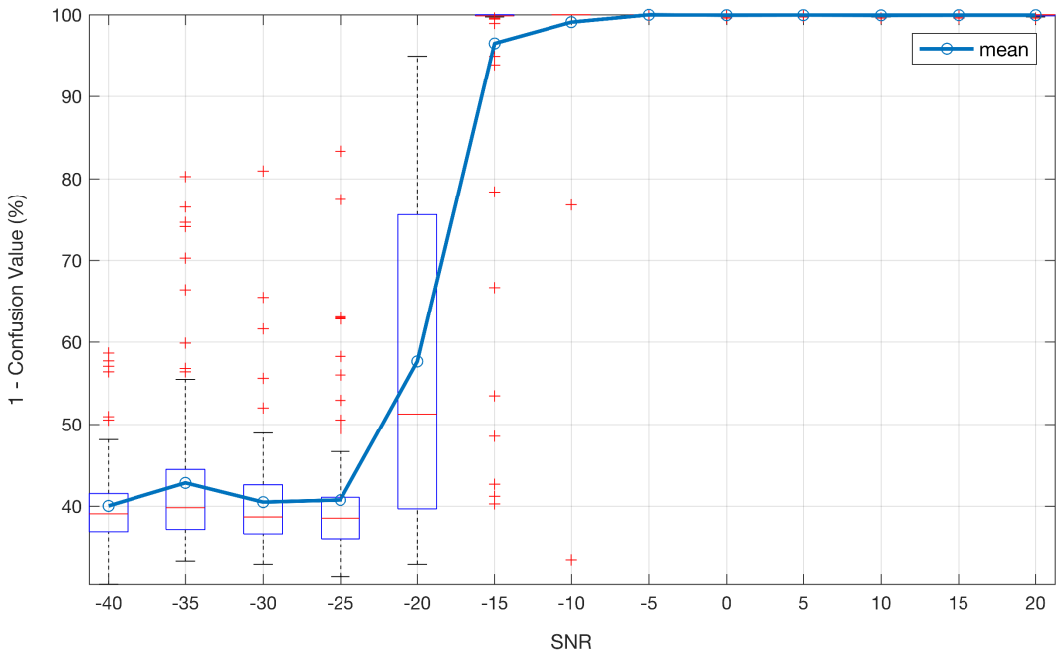


Fig. 3.10 Classification Performance of MLP Neural Net with 20 Hidden Neurons trained on Synchronized Time-domain Frames with SNR=[-40,20]

compared to the MF comes at the cost of first forming a synchronized dataset so that the time-domain RAT features appear at the same position consistently. However, one may argue that this pre-processing stage is not a real design cost, since almost all communication systems employ some sort of synchronization technique anyway.

Classification Performance Effect on CR Throughput

Finally, one way to connect the above performance values to CR throughput in terms of misclassification could be explained as follows. Looking at the throughput of generic primary system communication link from a SNR perspective and using the throughput definition used by the authors of [164], it can be shown that the throughput of a link is a cascaded multiplication of several probabilities as indicated by Equation 3.10.

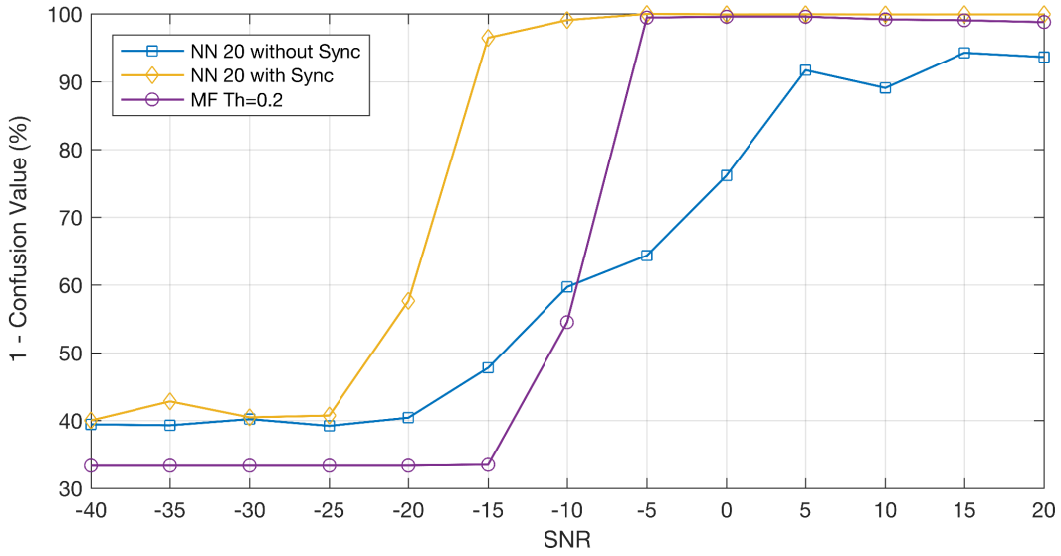


Fig. 3.11 Mean Correct Classification Percentage at various SNR levels for the three classifiers: Neural Net with Data Synchronization (yellow), Neural Net without Data Synchronization (blue), Baseline Matched Filter with Threshold=0.2 (purple)

Namely, the probability whether the transmitting node is actually transmitting at a particular moment, whether the receiver is silent and is not transmitting at this moment, and, finally, the probability for the communication channel to be regarded as a reliable link.

$$T = P\{probe\ transmits\}P\{receiver\ silent\}P\{no\ outage\} \tag{3.10}$$

The latter term fundamentally means,

$$P\{no\ outage\} = P\{SINR \geq \gamma^*\} \tag{3.11}$$

Where γ^* is a tolerated threshold that ensures reliability over a communication link. The SINR at the receiver needs to exceed γ^* for the received signal to be successfully recovered [164]. In the case of secondary-secondary cooperation

working in an underlay scenario the misclassification of a signal, i.e. a primary user to be classified as secondary user, will lead to an increased outage in the primary system, and hence a lower throughput.

The down side of this technique is that security aspects could be exploited by adversaries. Primary User Emulation [165], Connection attack [125], and Random Noise attacks [124] are a few to name. For this reason, unsupervised learning techniques, and specifically K-means clustering and Self-organizing maps (SOMs), have been researched to do the signal classification; in which case the classifier will have to tolerate little availability of a priori information [125]. Nevertheless, the MLP neural net technique from the implementation point of view was very simple and straightforward. It can easily be implemented for online mode operation. It is fast, where new inputs are just passed through the trained network and are classified with a class immediately.

3.5 Potential Applications

In what follows we will discuss use cases of this technique in the context of cognitive mobile spectrum sharing and enriching Radio Environmental Maps.

Cognitive collaboration in overlay CRNs

Considering adaptation mechanisms in overlay CRNs, when SUs recognize the type of the access technology used by the PUs they have the privilege of exploiting their traffic pattern so that they can fill up the gaps/silent instances of the PUs that they usually produce by using a particular RAT.

Secondary-secondary cooperation in shared spectrum

Another application is to reduce interference generated by the SUs affecting other SUs. In most of the published papers, authors assume that there are only PUs in a certain frequency band, e.g. TV Bands, and they try to fill up the white space in between PU channels while one can have more than one secondary network trying to transmit in the same white space band. To counteract situation alike, through RAT recognition an SU may observe the RAT used by another SU currently occupying a certain band, e.g. detection of a WiFi transmission (which is inherently a contention based multiple access scheme), and would want to contend for bandwidth. Even if the working SU is not currently using a contention based multiple access, both SU can negotiate a multiple access scheme, e.g. WiFi, so that both can use the band simultaneously.

3.6 Summary and Conclusion

This work has shown a simple yet efficient Multi-Layer Perceptron neural network classifier that is able to discriminate between three different radio access technologies in an automated supervised learning manner while consistently beating the baseline Matched Filter results. The MLP architecture achieves 99% correct classification at SNR=-10 dB and 97% at SNR=-15 dB, while the baseline method only achieves 55% at SNR=-10 dB and 33% at SNR=-15 dB.

Traditionally, multilayer neural nets have been used with manually selected feature inputs. However, in this chapter we presented an neural network architecture that relied least on predefined system parameters, within which we abstained from manually extracting features from the received input sig-

nals. Minimal signal pre-processing avoids over-parametrizing the classifier and make it as generic as possible so that it can be extended to other types of RATs. A testbed implementation has demonstrated the practicality of the research undertaken. Using such a classifier, secondary users will be able to sense whether a band is occupied by a primary or secondary user through detecting the type of radio access technologies received. The classifier can also be an enabler in the concept of coexistence between secondaries with different types of radio access technologies.

The performance limitations of MLP neural nets stem from the kinds of features used. If appropriate features are used, one can successfully train neural networks on very sophisticated datasets. However, if wrong features are used, perceptrons are extremely limited in terms of their learning capability. This has been verified via our experiments with the synchronized and non-synchronized RAT frames, where the same MLP architecture gives very different results the mentioned two cases.

In the next chapter, a novel Self-Organizing Map (SOM) and Support Vector Machine (SVM) based RAT signal clustering and classification scheme is presented for further enhancement of the classification performance.

Chapter 4

Radio Environment Feature Extraction and Classification using Self-Organizing Feature Maps and Support Vector Machines

4.1 Introduction

In the previous chapter we relied on the MLP neural network architecture to identify the type of the Radio Access Technology and we were able to achieve 97% correct classification at -15 dB SNR level. The research was aimed at scenarios were there may be coexistence of Primary Users and Secondary Users. We established that signal classification of the PU and other SU transmissions as one of the essential functions of Cognitive Radios. This is because while

RAT recognition enables a CR to avoid collision with PUs' communication, it also allows the SU CR to be aware of transmit opportunities *in contention* with other SUs. Otherwise, if channel occupancy detection was based only on power spectrum density (PSD) the CR would lose transmission opportunities under the assumption that a particular channel(s) is occupied by a PU.

This chapter, inline with the incentives of Chapter 3, aims to first demonstrate the suitability of a combined bespoke machine learning architecture for classification of different RAT frames, however in this case using frequency domain data. As shown in Figure 4.1, the proposed method in this chapter is comprised of a concatenation of an unsupervised Self-Organizing Feature Map (SOFM) neural network layer followed by a supervised Support Vector Machine (SVM) discriminator, forming a *semi–supervised* machine learning technique. Later, toward the end of the chapter we attempt to generalize the bespoke methodology of this work to the useful recognition of new RATs that have not been trained on before. Achieving such classification of different transmissions in *noisy environments*, will be essential to the realization of the CR paradigm, especially in the foreseen case of PU and SU cooperation and SU-SU coexistence which are currently highly regarded scenarios by the research community [166, 167].

The remainder of this chapter is organized as follows. Section 4.2 describes the data collection and pre-processing stages. Later, Section 4.3 discusses the machine learning techniques proposed for RAT classification as applied to the collected data. The results of the experiments are discussed in Section 4.4. Some of the potential application scenarios such as Radio Environmental Maps (REMs) are discussed in Section 4.5. Finally, a summary and some concluding remarks are provided in Section 4.6.

4.2 Frequency-domain Dataset

In order to replicate a practical scenario for the purpose of this research, a testbed was assembled as was previously presented in Section 2.3, and shown in Figure 2.8.

In Chapter 3 we used the raw received power, captured from the USRP in *time domain*, as input to the MLP neural network classifier. However, in this chapter, we intend to transform the data to the frequency domain based on the fact that some of the features are encoded in frequency domain by design, such as the pilot synchronization subcarriers. In this regard, the data samples pertaining to each noise level are transformed to the Frequency domain via Welch's Power Spectral Density (PSD) estimate [168], with a Hamming window. Welch's method can be simply described as the averaged periodograms of overlapped and windowed signal sections.

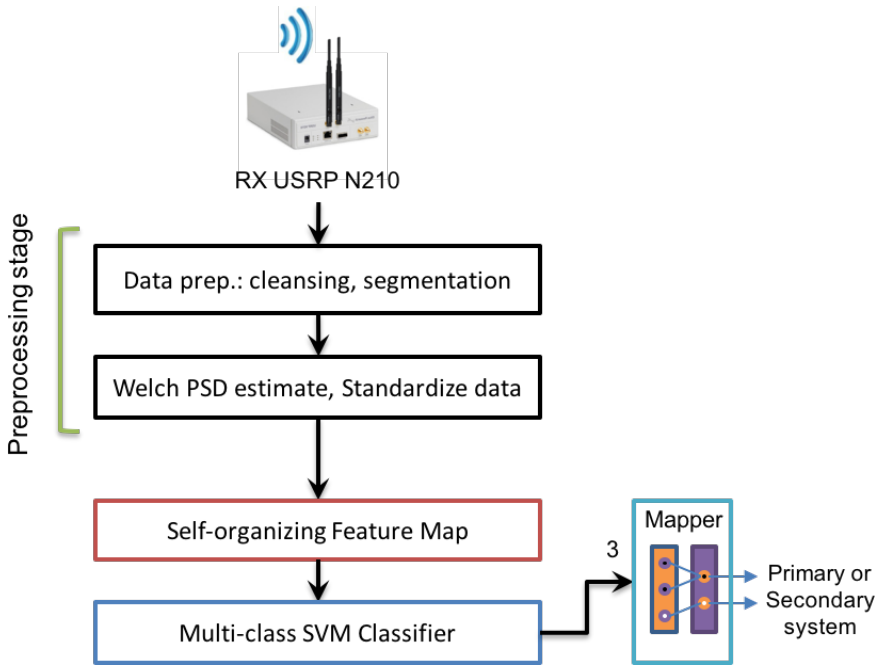


Fig. 4.1 Flowchart of the Proposed SOFM-SVM Method

Analytically, Welch's PSD can be described as follows. Let the mth window of a received time-domain frame be denoted by,

$$x_m(n) \triangleq w(n)x(n + mR), \tag{4.1}$$

where R is defined as the window hop size, K denotes the number of windows, $n = 0, 1, \ldots, M - 1$ and $m = 0, 1, \ldots, K - 1$. Then the periodogram of the mth window is given by,

$$P_{x_m,M}(\omega_k) = \frac{1}{M}|\mathrm{FFT}_{N,k}(x_m)|^2 \triangleq \frac{1}{M}\left|\sum_{n=0}^{N-1} x_m(n)e^{-j2\pi nk/N}\right|^2 \tag{4.2}$$

Finally Welch's PSD estimate is given by,

$$\hat{S}_x^W(\omega_k) \triangleq \frac{1}{K}\sum_{m=0}^{K-1} P_{x_m,M}(\omega_k) \tag{4.3}$$

It is worth mentioning that in line with our technique's obviation to system parameters, Welch's PSD estimation method is a non-parametric method too, where the PSD is estimated directly from the signal itself, avoiding possible modeling errors.

In order to display the performance of the SOFM-SVM technique, a stress test experiment has been developed. Under this experiment, a range of different levels of white Gaussian noise has been added to the original signal so that the SNR varies from 20 dB to -40 dB in steps of 5 dB. Figure 4.2 presents an instance of each of the considered RATs at different SNRs.

Finally, the data pre-processing stage ends by applying z-score to the data as

show in Equation 4.4.

$$z = \frac{x - \bar{x}}{\sigma} \tag{4.4}$$

z is a z-score of a spectrum data vector, x is the spectrum data vector, $\bar{x}$ and σ are the mean and standard deviation of that spectrum data vector, respectively. Essentially, in our research we regard each received frame as a single data vector. Thus our data pre-processing analysis is done on a vector-by-vector basis. In Data Science, this step is called *Data Standardization*, and it is particularly useful when analyzing data from different sources. The standardized data from all RAT sources have zero mean and unit standard deviation, with 95% of the data confined within -2 and +2 in magnitude. However, it retains the shape properties of the RAT data set, such as skewness and kurtosis measures, which are essential for the pattern recognition task at hand. Data standardization also helps in correcting for the different power levels received by the different USRPs.

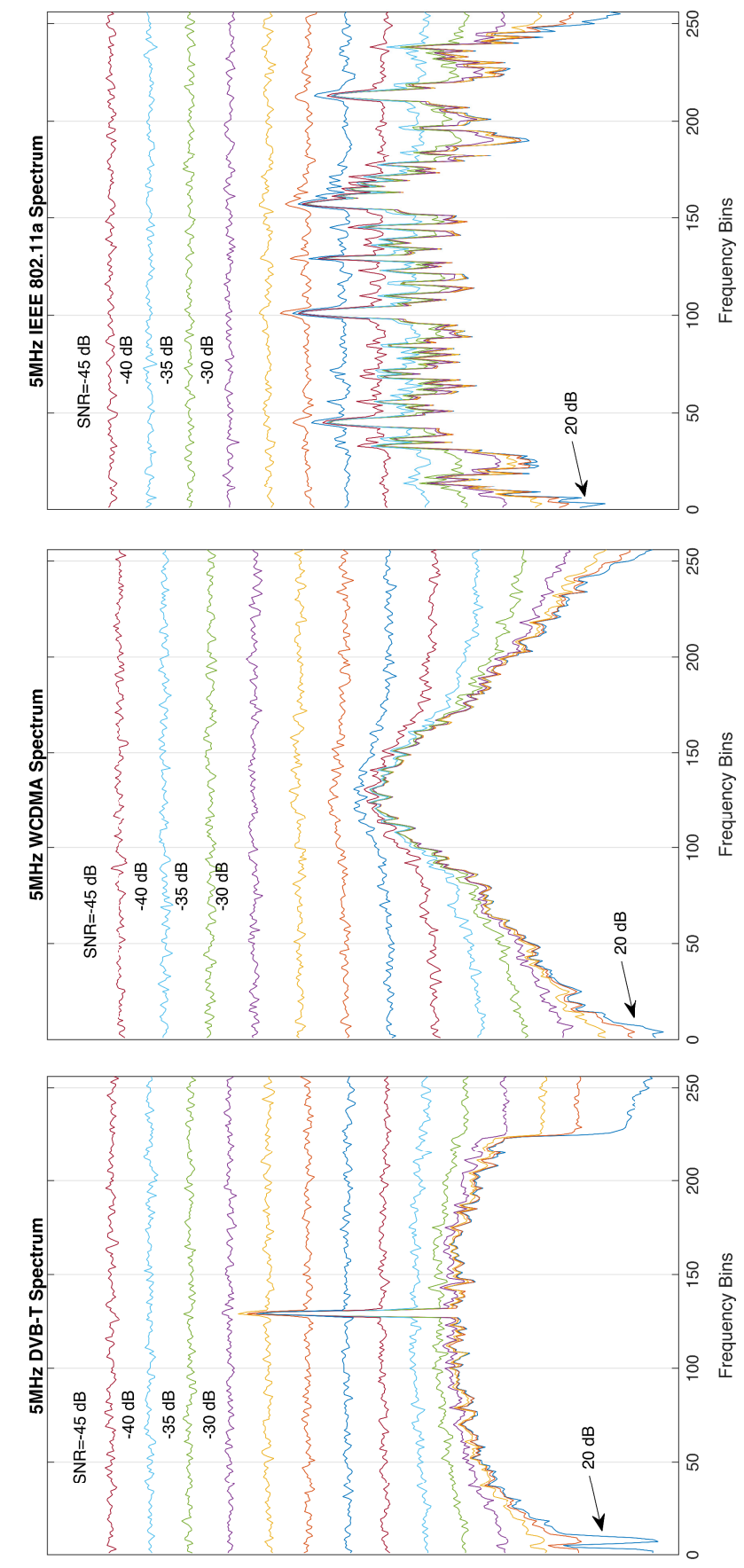


Fig. 4.2 Sample Frequency Spectrum of Three RATs: DVB-T, WCDMA, WiFi (Left to Right), with 14 Level of Artificially Added White Gaussian Noise

4.3 Self–Organizing Feature Maps and Support Vector Machines

In the following two subsections we will give a brief description of the two main machine learning techniques used in this chapter.

4.3.1 Self-Organizing Feature Map

The algorithm driving the Self-Organizing Feature Map (SOFM) makes use of the competitive learning concept, where small computation units, called "neurons", compete against each other such that only one neuron, the "winning neuron", is activated at a particular time. This results in neurons drifting to areas in the feature space that the sample vectors match most, and thus eventually forming clusters. This is illustrated in Figure 4.3, where the winning neuron (in black) is attracting the neighboring neurons towards itself.

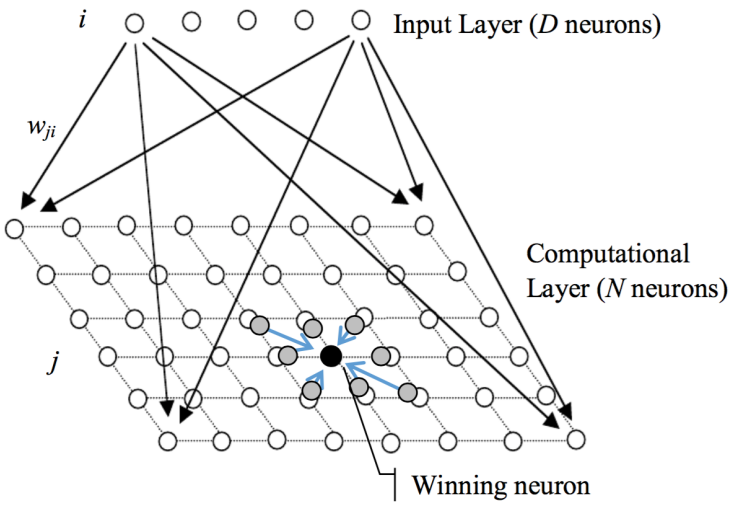


Fig. 4.3 Self-Organizing Feature Map indicating a the Input Layer, Computational Layer and the Winning Neuron. Note: the SOFM is layed out in *gridtop* topology here for illustration purposes only.

As shown in the Figure 4.3, the input layer has D dimensions where an input vector $x_i = \{x_i : i = 1, \ \ldots, D\}$, and the weights of the connections between each input neuron and the output computational layer can be written as $w_{ji} = \{w_{\text{ji}} : j = 1, \ldots, N; i = 1, \ldots, D\}$, where N is the total number of neurons at the computational layer. The SOFM algorithm iterates the following steps until the feature map is established:

1) Initialization: initialize the values of the weight vectors w_{ji} in a random manner.

2) Sampling: randomly draw a data vector x_i from the input space.

3) Similarity Matching: calculate the discriminant function,

$$d_j\left(x_i\right) = \sum_{i=1}^{D} \left(x_i - w_{ji}\right)^2 , \tag{4.5}$$

which is the sum of squared Euclidean distance, and then de-map the index winning neuron $I(x_i)$.

4) Updating: update the weight vectors by applying the weight update equation,

$$w_{ji} = \eta\left(t\right) \ T_{j,I(x_i)}\left(t\right) \ \left(x_i - w_{ji}\right) , \tag{4.6}$$

where $\eta\left(t\right)$ is the learning rate where $0 < \eta < 1$, and $T_{j,I(x_i)}\left(t\right)$ is a Gaussian Neighborhood function,

$$T_{j,I(x_i)}\left(t\right) = e^{-\frac{S^2_{j,I(x_i)}}{2\sigma^2}}. \tag{4.7}$$

$S^2_{j,I(x_i)}$ is the lateral distance between an input neuron i and an output neuron j. In order to get definite clusters, the size of the clusters $\sigma(t)$ needs to shrink with time. A well-known distance decay function is the

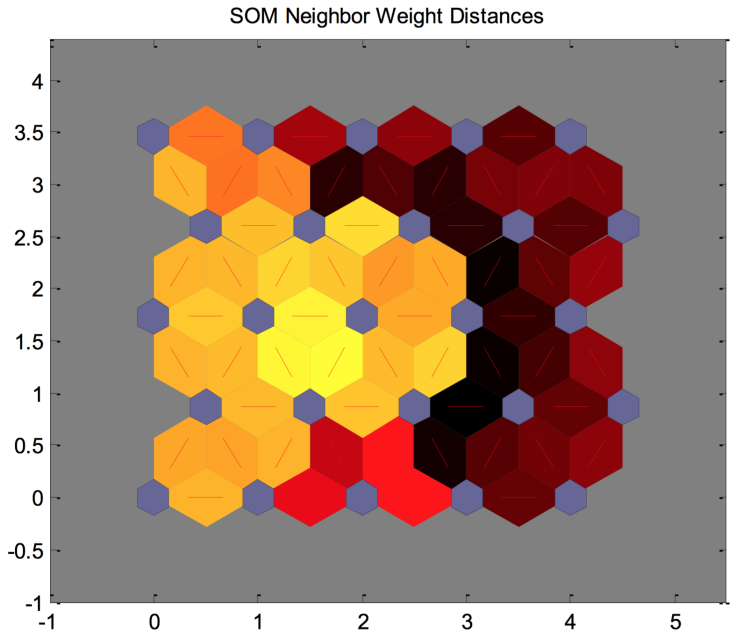


Fig. 4.4 SOFM Neighbor Weight Distances, SNR=20 dB

exponential decay function,

$$\sigma\left(t\right) = \sigma_0 \; e^{-\frac{t}{\tau_\sigma}}, \tag{4.8}$$

where σ_0 is the initial neighborhood distance at $t = 0$, and τ_σ is the mean lifetime of the size of the neighborhood decay. SOFM neighborhood distances are shown in Figure 4.4, where light and dark colored neural connections indicate closely situated further apart neurons. Similarly, the learning rate shall also decay in order for the algorithm to converge and stabilize.

5) Continuation: return to step 2 until the feature map is stable.

SOFM Simulation Parameters

The SOFM has a peculiar neural network architecture. Unlike normal MLP neural nets, the weight matrix in a SOFM forms the feature map, which is the intended output of this clustering technique. Based on extensive experimentation, the essential parameters involved in configuring and training the weight matrix of the SOFM are described as follows:

- The similarity distance function, as discussed above, is the negative Euclidean distance z between the input vector $p \in P$ and the corresponding neuron weights w, which can be described as,

$$z = -\sqrt{\sum_{P}(w-p)^2} \tag{4.9}$$

- The initial neighborhood size is set to 3.
- The chosen topology of the SOFM is *hextop*, which defines the pattern of the neurons surrounding the winning neuron as shown in Figure 4.4.
- Number of steps for neighborhood to shrink to 1 (τ_σ)= 100.
- Number of neurons in the computational layer = $6 \times 6 = 36$ neurons, with the number of weights = 6400.
- Input weight initialization function: Principal Component Initialization. This method initializes the weights of an N-dimensional self-organizing feature map so that the initial weights are distributed across the space spanned by the most significant N principal components of the inputs. This leads to a significant speed up in the SOFM training process [92].
- The SOFM training algorithm is based on batch unsupervised weight/bias training. The maximum number of training epochs = 1000.

4.3.2 Training the SVM Layer

Having trained the SOFM layer, in this step, pairwise combinations of the neuronal clusters are fed into an SVM model, where each neuron from the SOFM layer is weighted according to the number of times it was chosen as the winning neuron, or simply said the number of "sample hits" it has received during the SOFM training period. With the aid of the Kernel trick, SVMs can perform non-linear classification efficiently [169]. In this work a Radial Basis Function has been applied as the kernel function. A brief analytical introduction to SVMs Primal Formulation has been discussed in Section 2.2.5. Below we discuss nonlinear transformation with the help of the kernel trick. However, for a more comprehensive treatment of SVMs' background review, setup and training, the reader is kindly suggested to refer to [96].

Nonlinear Transformation with Kernels

There is a solution to avoid training complicated hyperplanes in binary classification problems. Using the *Kernel Trick* [170] we can transform observations to another space where we can use linear classification – as if the data was linearly separable.

This approach uses these results from the theory of reproducing kernels:

- For two data points x and x', there is a class of functions $K(x, x')$ with the following property. There is a linear space S and a function φ mapping x and x' to S such that

$$K\left(x, x'\right) = \left\langle \varphi\left(x\right), \varphi\left(x'\right) \right\rangle \tag{4.10}$$

  The dot product $\left\langle \varphi\left(x\right), \varphi\left(x'\right) \right\rangle$ takes place in the space S.

- The class of functions used in this research is the Gaussian Radial Basis Function (RBF), where for some positive number σ,

$$K(x, x') = \exp\left(-\frac{\langle\varphi(x), \varphi(x')\rangle}{2\sigma^2}\right) \tag{4.11}$$

  where $\langle\varphi(x), \varphi(x')\rangle$ is the squared Eclidean-distance $\| x - x' \|^2$ between x and x'.

Dot products are at the core of classification hyperplane calculation. In order to built non-linear classifiers, nonlinear kernels can also use the same technique. Where in this case the we will obtain hyper-surfaces in some space S, instead.

Kernel Selection Procedure

In order to optimize for the most suitable kernel function and parameters, the following steps have been taken in designing the SVM model [92]:

1. Train a model using SVM, starting with $C = 1$ and scaling factor $\sigma = 1$
2. Cross-validate the trained model in step 1, with the cross-validation factor k=10.
3. Use the model to classify (predict) new data.
4. Tune the type of the kernel function and its parameters.

In this Kernel tuning exercise, we considered the 'RBF' and the 'linear' type of kernels. As for the Hamming window used in Welch's method, as was mentioned in Section 4.2, we considered a set of window lengths ranging from 300 to 900 bins. As a result of an extensive set of simulation runs, we found out that the Hamming Window of 600 bins with either Linear or the RBF kernel was achieving better classification results than the rest of the configurations.

4.4 Results and Discussion

To demonstrate the potential of the SOFM clustering technique, Figure 4.5 shows the SOFM Sample Hits diagram at SNR = 20 dB, where the size of the hexagons indicate the relative number of sample hits at each neuron. The hexagon with 249 sample hits, shown in the middle of figure, represents the exact 249 training samples from the recorded WCDMA transmissions that were fed into the SOFM. The sample hits on the right and left belong to the other two RATs. The empty space between these regions is a clear fundamental indication that there were no confusing hits between the clusters. In other words, if we were to draw separating boundary, 100% correct clustering of the different RATs can be achieved.

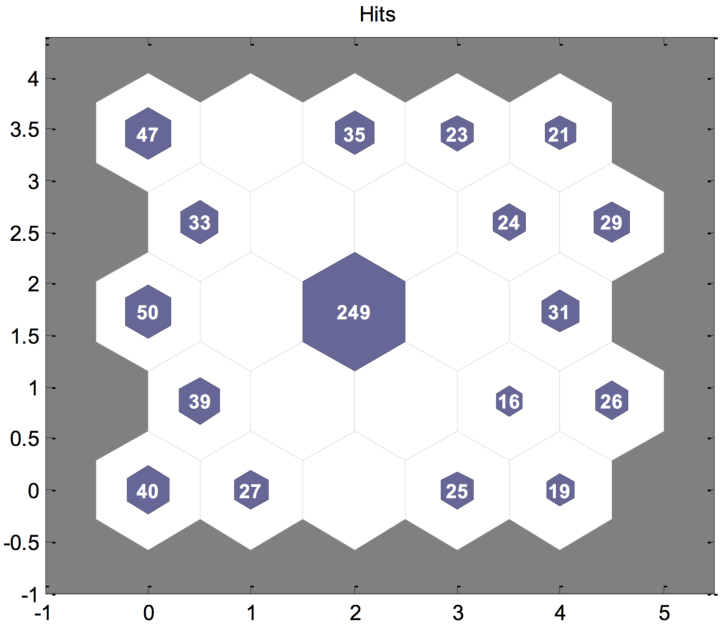


Fig. 4.5 SOFM Sample Hits, SNR=20 dB

In order to test the performance of the SOFM architecture in a noisy signal scenario, white Gaussian noise has been added to the signal to lower the SNR.

The performance for the resulting new adapted data set, at 0 dB SNR, is shown in another simulation run in Figure 4.6. Here, dark-red colored inter-neuronal links indicate that such pair of neurons are situated far apart in the SOM feature space, and the vice-versa for the dark-blue colored links. In this experiment we can here observe that non-linear classification will be an essential requirement since as we add noise to the received data frames the SOFM neuronal clusters eventually starts to overlap and become linearly inseparable.

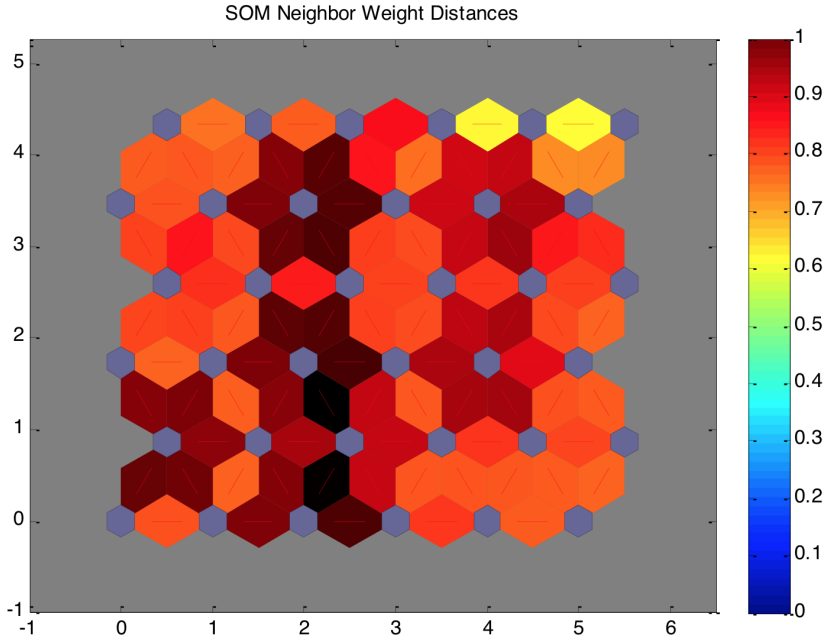


Fig. 4.6 SOFM Neighbor Weight Distances, SNR=0 dB

In Figure 4.7 we show the SOFM sample hits diagram with the decision boundaries drawn by the SVM. Due to the considerable amount of noise added, around 20 dB, the decision boundaries between the different RAT clusters have been fused to some extent. Note that the number of neurons in the computation layer of the SOFM is also increased from $5 \times 5 = 25$ to $6 \times 6 = 36$

neurons. This way the number of hits that each neuron gets becomes relatively less, and thus allowing for a higher resolution of the neuronal hits.

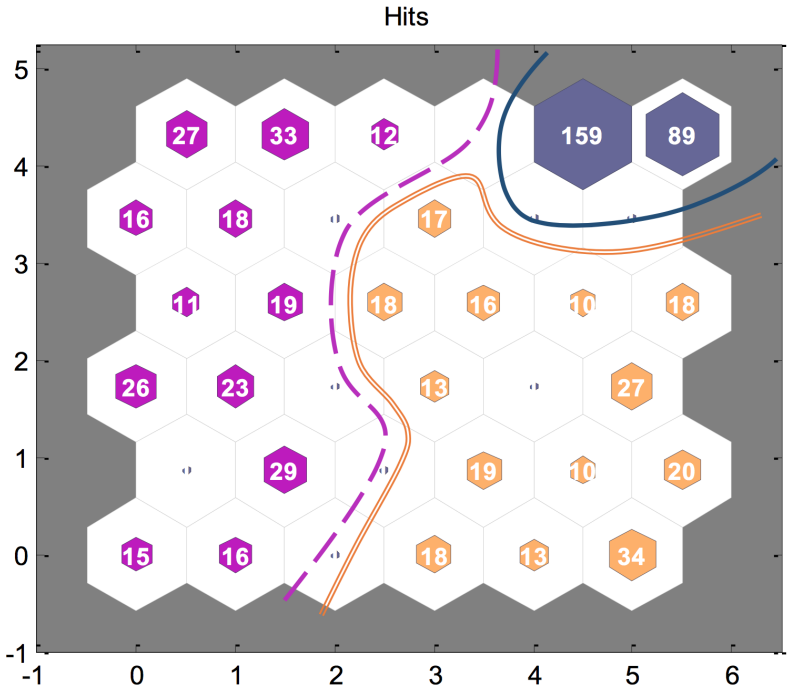


Fig. 4.7 SOFM Sample Hits, SNR=0 dB. Shown in different colors are the 3 RAT clusters that have been separated from each other the SVM decision boundaries

In order to benchmark the performance of the proposed SOFM-SVN method, Matched Filters are used as the baseline performance measure. The spectral kernels used here were built from taking an average of the highest 20 dB SNR spectra of each RAT as shown in Figure 4.2. This is also the same methodology used to build the time-domain kernels in Chapter 1

Figure 4.8 shows classification results using matched filters with different threshold values ranging from 0.75 to 0.9 in steps of 0.05. Note that although the MF configuration with threshold equal to 0.75 provides the best classification results at SNR=-10 and -5 dB, it performs poorly at 15 and 20 dB. The lower

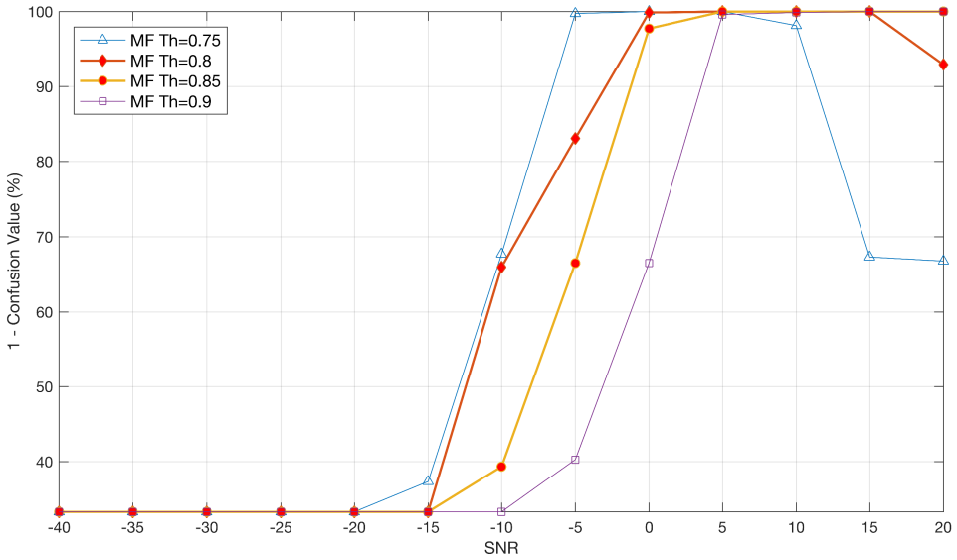


Fig. 4.8 Performance of SOFM-SVM technique under various SNRs

performance at -10 and -5 dB SNR is due to the fact that, when the threshold level is lowered, more than one RAT passes the threshold value when filtered with the spectral kernels used. This leads to an increase in false-alarm probability and thus a higher confusion value. As for the MF with threshold value equal to 0.8, we observe a similar performance to that of threshold=0.7 at SNR=-10 dB, but we are trading off lower classification performance at SNR=-5 dB for relatively higher performance at SNR=20 dB. Therefore, we choose the MF with 0.8 threshold value as the candidate MF for comparison with the proposed method as discussed below.

Figure 4.9 presents the final classification performance result of the proposed SOFM-SVN method. In our research in the beginning we considered two types of kernels: Linear And Radial Bases Function (RBF). However, we found out that their results were not significantly different from each other. A possible explanation of this case could be that the SOFM clusters were linearly separable.

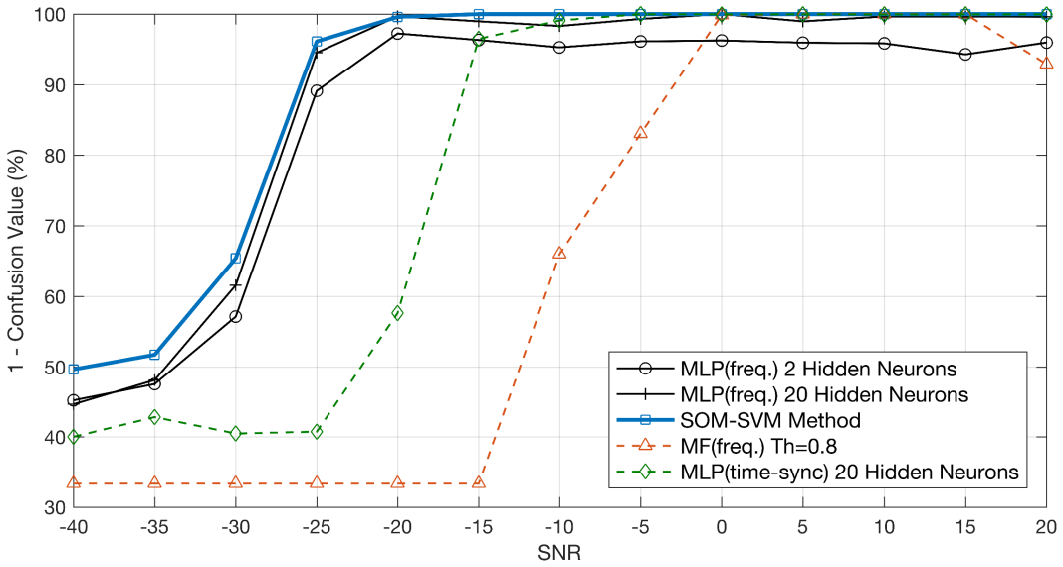


Fig. 4.9 Performance of SOFM-SVM technique under various SNRs ranging from -40 dB to 20 dB, as compared to 3 MLP neural network classifiers with 2 and 20 hidden neurons (the latter being trained on both frequency and synchronized time-domain data as discussed in Chapter 3), and a matched filter classifier with threshold value=0.8 from Figure 4.8

Note that the final result of the SOFM-SVM method shown in Figure 4.9 in blue with square markers is the average of the binary classification performance combinations between all possible pairs of RATs. In other words, we have taken the average classification performance value for DVB-T vs. WCDMA, DVB-T vs. WiFi and WCDMA vs. WiFi averaged at every SNR level.

In Chapter 3 we had considered an MLP architecture with 20 hidden neurons trained over synchronized time-domain transmission frames. In order to compare the research done in this chapter to that of the previous, we have again trained an MLP neural net with 20 hidden neurons, however, unlike before, the training here is done on frequency spectrum data. The synchronization stage introduced previously is also not necessary here since the considered PSD estimate of the RAT features, i.e. the preamble and pilot signals, stays constant in terms of its position in the train data vector. The dashed green line in Fig-

ure 4.9 with diamond markers represents the previously trained MLP neural net with 20 hidden neurons in time-domain, which is the same classification result gotten from Chapter 3, included here for the purpose of comparison.

4.5 Potential Application Scenarios

The RAT recognition technique discussed in this chapter has multiple potential applications. Such a technique could be part of SU sensing-enhanced White Space Database (WSDB) access [5], augmenting the capabilities achieved under current regulations (see, e.g., [171]). In the following subsections we will elaborate more on the above mentioned potential case scenarios.

4.5.1 Radio Environmental Map

Ofcom allows qualifying database operators to provide TVWS database service [4, 5]. Figure 4.10 shows a schematic representation of the plan to adopt WSDBs by Ofcom in the UK.

Secondary users that would wish to be authorized to use TVWSs need to initially choose a database operator from a list of qualifying database operators provided by Ofcom [4]. Broadly speaking, each SU needs to inform the database about its location and provide other information allowing the database to identify its radio characteristics. In response, the WSDB will reply to the SU with the currently authorized frequencies and power levels, and an expiry time of this authorization [4, 5]. WSDB operators can in effect enhance their database service provision with Radio Environmental Map (REM) functionality. REMs, in the cognitive radio networking framework, may track the locations and activities of SUs, act as policy enforcing nodes, and be the

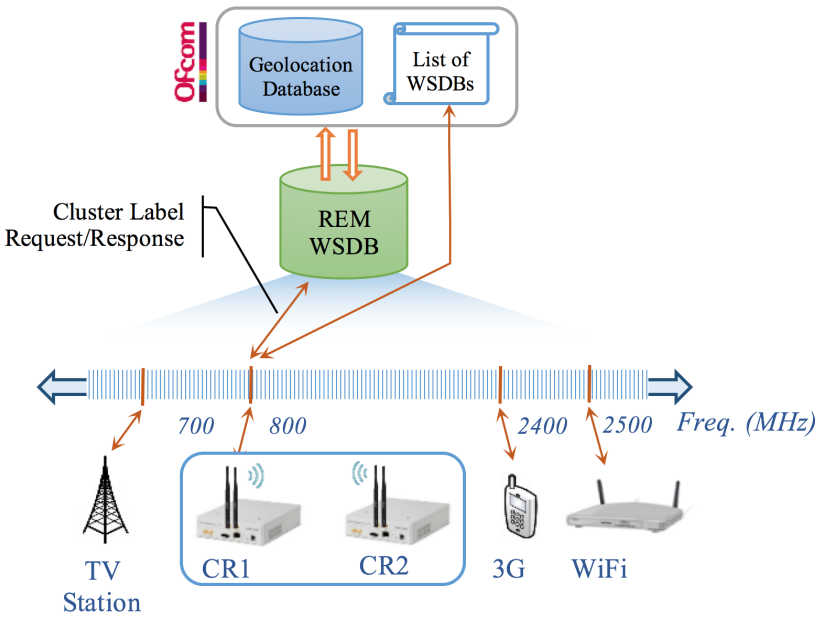


Fig. 4.10 A schematic showing Ofcom's plan to adopt WSDBs

main point for storage of common geographical information.

In this context, we believe that integrating RAT classification functionality into REM-enabled WSDBs could add to their value. One example of this is to assist secondary-secondary coexistence [166]. It might commonly be the case that many secondary devices are equipped with a range of different qualifying white space standards, which might be chosen from for white space access with that choice being unknown to the WSDB. A REM-enabled WSDB, having recognized the chosen characteristics of particular white space devices (WSDs) within each other's vicinity, might be aware of coexistence issues between the WSDs and might adjust allowable transmission powers accordingly. Such an option is, nevertheless, outside of regulatory and WSDB scope and would require regulatory enhancements to be implementable.

Such REM-enabled WSDB capability might also even be beneficial for the conventional primary-secondary coexistence purpose of WSDBs. It is noted that

aggregate interference issues of WSDBs have generally not been sufficiently embedded in current proposed regulations. A WSDB, through detecting information on the utilized RATs and extent of utilization of them by authorized WSDs, might detect the potential for aggregate interference to become a problem to primaries, and adapt authorized powers to WSDs accordingly.

4.5.2 Learning By Collaboration

The SOFM-SVM techniques' functioning could be to some extent analogous to how we, humans, classify things around us. For instance, when a new item is for the first time presented to us, we instantly categorize it according to its pertinent physical properties. Then we either ask for its name or call it a name if it was discovered or invented. Thereon, mentioning the new objects name will be sufficient to remember its properties.

In the same manner, we put forward the following two scenarios on how a RAT classification in a cognitive radio networking context could take place. In case we already have labeled training data, then after clustering the data via the SOFM technique we can basically map the data to their labels and classify the received frames by using SVMs or MLP NNs. On the other hand, in some situations we may not have labeled data – which corresponds to the case where a CR receiving transmissions that it does not know to which standard it belongs. The SOFM algorithm will be able to create a new cluster for this new type of RAT, however, it will not be able to recognize it since it has not seen this type of RAT yet. The solution to such a situation could be such CRs will have the capability to send multiple samples of this newly discovered RAT or their operating frequencies to the REM WSDB as shown in Figure 4.10. These REM-based WSDBs are generally presumed to have a high

computational power compared to secondary CRs. They are also the most knowledgeable of the devices currently in the area, their frequencies and the type of RATs they use. After classifying the received samples from the CR or looking up the type of RAT inhibiting the inquired frequency band, the REM database can then respond to the CR with the requested information.

Considering the case of Docitive Networks [76], where intelligent CRs may wish to teach or learn from one another, it is also worth mentioning that no two communication devices necessarily possess the same radio characteristics and hardware performance – e.g. dynamic range, phase noise, noise floor. Thus, in this context, device configuration exchange may not lead to enhanced performance in scenarios where CRs could copy other CRs internal configurations. This could be especially the case with current SDR equipment: each of which feature diverse technical characteristics. Thus, inter-CR-device configuration transfer, e.g. in the realm of Docitive Networks concept, may turn out to be difficult. CRs may therefore need to build their own optimized cognitive engine configuration; which directly depends on how much noise they can tolerate, how many neurons in their cognitive engine's neural network is computationally feasible, how much CPU (Central Processing Unit) power and RAM (Random Access Memory) they carry on board, and other technical features alike. Henceforth, we believe that semi-supervised techniques similar to the SOFM–SVM technique discussed in this chapter shall present a good first step forward.

4.6 Summary and Conclusion

In the previous chapter, our radio access technology classification analysis was based on time-domain signals collected from the USRPs. In this chapter

we moved on to consider the frequency domain spectrum which has shown a higher classification performance using the proposed SOFM–SVM classifier to discriminate between three different RATs. Using the SOFM–SVM technique, secondary users can sense whether a band is occupied by a primary or another secondary user through detecting their transmission protocols, in addition to alternative possible benefits, such as the application of this concept in REMs used to enrich WSDB capabilities. This approach has been compared with a fully supervised Matched Filter and MLP neural net classifiers in frequency domain and with the MLP neural net classifier in time-domain, as was previously discussed in Chapter 3. The SOFM-SVM method consistently achieves superior performance than the above mentioned methods, as was presented in Figure 4.9, where it is 8% better classification performance at -30 dB and below, and at least the same or better performance at -25 dB and above.[140, 172] Apart from the superior classification performance the proposed SOFM-SVM technique has the advantage getting rid of the frame synchronization stage we had introduced the previous chapter. In this chapter, however, we had to transform the received data from time to frequency domain. This was achieved using Welch's PSD estimate which was optimized over a set of hamming window periods.

Regarding the challenges encountered in developing the SOFM part, in order to better locate each neuronal cluster on the computational layer of the SOFM, we had to increase the number of neurons. This, of course, come at an increased computational cost to execute such an algorithm. In SVM's case, the Scale and C parameters had to be optimized.

Chapter 5

Radio Access Technology-Aware Cognitive Radio Spectrum Access: Using Deep Neural Nets and Reinforcement Learning

5.1 Introduction

For cognitive radios to access the TV White Space (TVWS) spectrum, they need to be aware of the frequency spectrum for transmission opportunities periodically. It will be crucial for the cognitive radios to retain their previous experience when scanning different spectrum bands. Storing the spectrum availability pattern at different locations could be an additional cost. However, looking this information up periodically in order to find appropriate bands of spectrum will save a cognitive radio a certain amount of time. Moreover, when the TV white space spectrum can be scanned on a prioritized manner, the time

spent on spectrum scanning can be further reduced.

One way to accomplish this prioritization is to differentiate between TVWS channels[1] according to their license type, i.e. Primary Users (PUs) and Secondary Users (SUs). When a channel is occupied by a PU, access to such a channel is prohibited. However, if a channel is used by a SU, other SUs can contend for access. This is based on the fact that regulators protect PUs against interference [4, 173], while there is regarding restriction on interference on SU communications. Considering Dynamic Spectrum Access [12, 65] in such HetNets, a cognitive radio should have the ability to classify different RATs according to their usage license of the frequency spectrum, i.e. whether a RAT is a primary signal or a secondary signal.

Due to the various applications that require wireless connectivity, today we can find diverse combinations of Radio Access Technologies (RATs), which have led to Heterogeneous Networks (HetNets) [174]. It is it foreseen that in the coming 5G radio communication systems, various wireless technologies will be working side–by–side to complementing each other's functionalities to form a fast and efficient networking experience. For instance, one of the upcoming last mile wireless technology integrations is the combination of mobile and home networking. Within the 5G paradigm, an 802.11x home networking technology is going to be coordinating with the mobile connection of the members of that home [16, 30]. For cognitive radios to be part of such envisaged 5G HetNet they need to be able to recognize different RAT in their proximity [12], such that handover can take place between the cellular and home networks seamlessly. In this sense, here we consider a mixed primary–secondary HetNet where each RAT represents a particular licensing group. Here, we will consider Digital Video Broadcast–Terrestrial (DVB-T) and Wideband Code Division Multiple

[1]For the sake of brevity, hereinafter we refer to 'TVWS Channels' as 'Channels' and use them interchangeably.

Access (WCDMA) to represent primary user transmissions, and 802.11–based WiFi signal to be a secondary user transmission.

In the previous chapters we have presented machine learning-based RAT classification techniques that perform well under various levels of AWGN. This chapter presents yet another RAT classification technique, however, most importantly, it shows that RAT type recognition can be an important factor when used as contextual information for dynamic channel sensing and allocation.

The rest of the chapter is organized as follows. In Section 5.2, the system model and problem formulation in terms of combined Auto-Encoder NN and Reinforcement Learning algorithm; relevant features are obtained from the NN architecture as will be discussed in 5.3. Later, in Section 5.4, the extracted feature, along with others features, are used by a type of RL with function approximation. The goal of the chapter in to propose a novel algorithm that is implements a resilient and context-aware cognitive radio. In this regard, to test the suitability of the proposed algorithm for operation as an intelligent radio, Section 5.5 presents extensive simulation to evaluate the various performance metrics relevant to CR.

5.2 System Model and Problem Formulation

In this section, we will provide the system model for the simulation scenarios that will be discussed in the next section. In this chapter, we model a Cognitive Radio as a single agent trying to maximize its performance opportunistically, while respecting spectrum regulators' rules. The spectrum sensing and access will be done via Reinforcement Learning using the contextual information extracted by the DNN.

Herein, the operation of the DNN-RL algorithm is modeled in a dynamic radio environment, where:

1. The spectrum is divided into n channels of equal bandwidth, 8MHz as per UK TVWS [5].
2. The state s of a CR is the TVWS channel that it occupies.
3. The action a taken by a CR to access channel, is a deterministic process, where the agent will always land in the channel that was attempted to access. Sometimes, action a results in a collision if a primary was already using that channel, otherwise it is considered a benign transmission.

A flow chart of the proposed algorithm that will govern the functioning of the Cognitive Radio is shown in Figure 5.1, and its steps can be enumerated as follows:

1. Receive the spectral information from the interested portion of the spectrum band
2. Preprocess the data: cleansing, segmentation
3. Classify the transmissions as Secondaries or Primaries through a Deep Learning NN
4. Take an action to access a channel using reinforcement learning based on the labeled spectrum data (primary or secondary)
5. Receive reward from the radio environment

For the purpose of this chapter we shall consider a few different RATs coexisting in a contiguous chunk of spectrum. In our experiments, RATs pertaining to our hypothetical but realistic radio environment, where dynamic spectrum access is considered, are indicated in Table 5.1. The licensing information extracted

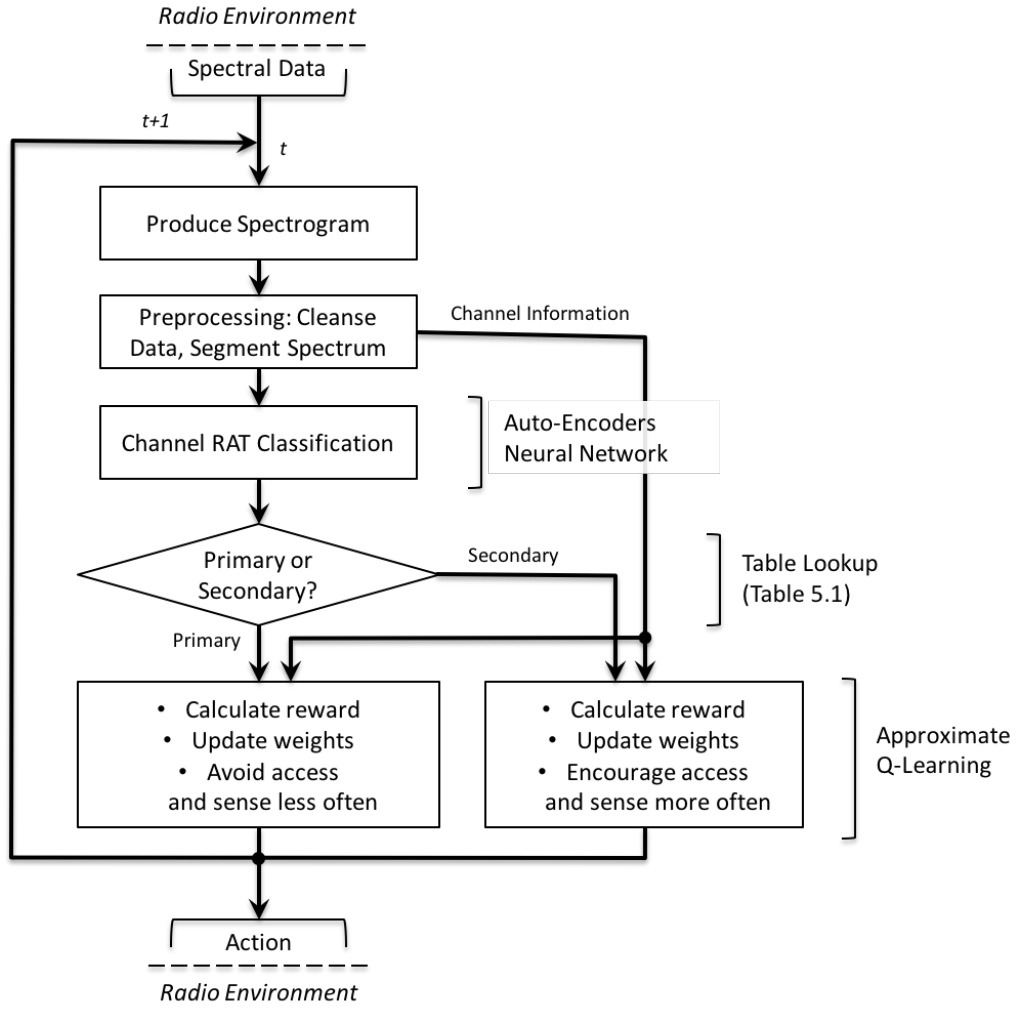


Fig. 5.1 Deep Neural Network and Reinforcement Learning based Dynamic Spectrum Access Flowchart

from the deep learner, based on the RAT types discussed above, is utilized by the reinforcement learning algorithm to avoid colliding with the primary users and exploit the less occupied frequency bands.

In Chapter 2, we discussed the different types of deep learning architectures. Here, we show how a DNN infers the licensing information of the incoming transmissions as either primary or secondary signals.

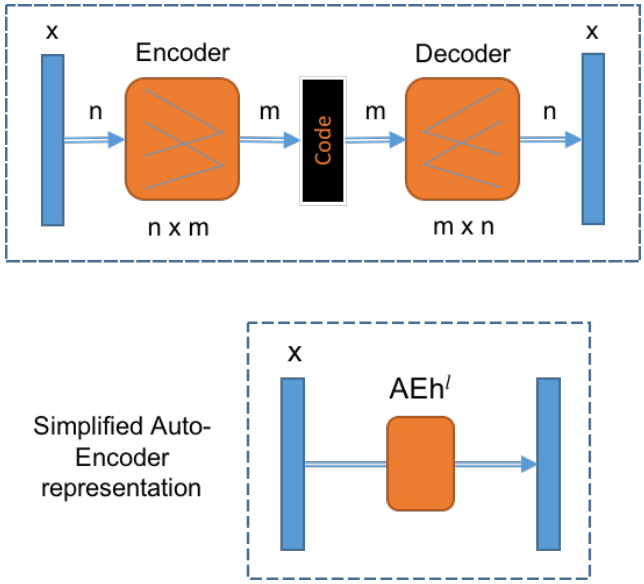


Fig. 5.2 A One Layer Auto-Encoder Neural Network

In this work we will consider Sparse Auto-Encoder Neural Nets [108] with regularized weight decay and sparsity. An architecture of an Auto–Encoder is given in Figure 5.2.

5.3 Sparse Auto-Encoders

This section is based on the Neural Network foundations developed in Chapter 2. A detailed description of Sparse Auto-Encoders (SAEs) can be found in [108].

Similar to MLP-Neural Nets, the cost function $J(\theta)$ of the model considered

Table 5.1 RATs used and their corresponding categories

802.11 WiFi	Secondary System
WCDMA	Primary System
DVB–T	Primary System

in this work is given by Equation 5.1, where:

- x is a training example indexed by $n \in N$, where N is total number of training examples
- $\hat{x}$ is the reconstructed version of the input example x
- i is the index of elements in an input x training example
- j is the index of the hidden units in the RBM layers
- l is the index of an RBM layer, $l \in L$
- λ is Weight Decay Regularizer
- w_{ij} is the weight on the link between input i and hidden neuron j.
- β is the sparsity regularization coefficient
- KL is the Kullback–Leibler divergence and its given by Equation 5.2, where:
  - ρ is the desired mean activation, $\rho \ll 1$
  - $\vec{p}_j^{\,l}$ is the mean probability of hidden unit j in layer l being active, as given by Equation 5.3

$$J(\theta) = \underbrace{\frac{1}{2N}\sum_{n}^{N}\sum_{i}(x_i^{(n)} - \hat{x}_i^{(n)})^2}_{MSE\ Error} + \lambda \underbrace{\frac{1}{2}\sum_{l}\sum_{i}\sum_{j}(w_{ij}^{(l)})^2}_{L^2\ Weight\ Decay} + \beta \underbrace{\sum_{l}\sum_{j}\mathrm{KL}(\rho \parallel \vec{p}_j^{\,l})}_{Sparsity\ Penalty} \tag{5.1}$$

$$\mathrm{KL}(\rho \parallel \vec{p}_j^{\,l}) = \rho \log(\frac{\rho}{p_j^l}) + (1-\rho)\log(\frac{1-\rho}{1-p_j^l}) \tag{5.2}$$

$$\bar{p}_j^l = \frac{1}{N} \sum_{i=1}^{N} h_j^l \tag{5.3}$$

The first term in Equation 5.1 is the error term which is modeled as the Mean Square Error (MSE) difference between the raw input and be reconstructed version as discussed previously. The second term is a weight decay [128] applied to counteract over-fitting, governed by the Weight Decay Regularizer λ. This term is also sometimes called L^2 *Regularization*. This regularization factor tries to prevent high weight values which may consequently lead to a low value at the output of a hidden unit and thus disguise as sparsity. Minimizing the error function $J(\theta)$ ensures high wights to be penalized. The third term is a sparsity penalty measure using the Kullback–Leibler (KL) divergence. Here, KL is used to calculate similarity between the distribution of a hidden neuron's activation, $\bar{p}_j^l$, to that of the desired mean activation distribution ρ. In other words, if $\bar{p}_j^l$ is equal to ρ then this term will be zero, otherwise a penalty will be added to the cost function.

Softmax Layer

The penultimate layer of the stack auto-encoder neural network is a Softmax layer. The purpose of this layer is to calculate the probability of the output being associated with a particular class given the input x, $P(y = k|x)$. A Softmax layer outputs a K-dimensional vector, where, upon a correct classification, the k^{th} index of the output vector should be the highest in value among

others. Analytically, a Softmax layer can be described as follows [175]:

$$h_\theta(x) = \begin{bmatrix} P(y=1|x;\theta) \\ P(y=2|x;\theta) \\ \vdots \\ P(y=K|x;\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x)} \begin{bmatrix} \exp(\theta^{(1)\top} x) \\ \exp(\theta^{(2)\top} x) \\ \vdots \\ \exp(\theta^{(K)\top} x) \end{bmatrix} \tag{5.4}$$

which is also equivalent to,

$$P(y^{(i)} = k|x^{(i)};\theta) = \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})}. \tag{5.5}$$

Note that the $\frac{1}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x)}$ normalizes the exponential in the numerator and therefore we get a probability function. Here, the model parameters are $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(K)} \in \Re^n$. For training a Softmax layer we will minimize the cost function,

$$J(\theta) = -\left[\sum_{i=1}^{m}\sum_{k=1}^{K} \mathbb{I}\left\{y^{(i)} = k\right\} \log \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})}\right] \tag{5.6}$$

Since we cannot minimize the equation above analytically, we will be using an iterative optimization algorithm. In this regard, we will be using the stochastic conjugate gradient descent. Taking the gradient of the equation above we get,

$$\nabla_{\theta^{(k)}} J(\theta) = -\sum_{i=1}^{m} \left[x^{(i)} \left(\mathbb{I}\{y^{(i)} = k\} - P(y^{(i)} = k|x^{(i)};\theta)\right)\right] \tag{5.7}$$

where $\mathbb{I}$ is the Identity Function.

Stacked Auto-Encoder Training Methodology

Training an Auto-Encoder may involve a data preprocessing stage for better classification performance. Since an Auto-Encoder tries to reconstruct its output based on its input, it is necessary that the raw input data match the range of the Encoder/Decoder Transfer Functions, i.e. in this case, *logsig* function, which ranges from 0 to 1.

Algorithm 2 Stacked Auto-Encoder Neural Network Training

Require: λ, β, ρ, D

1: **Preprocessing:** Divide raw data D into training D_{train} and testing D_{test} sets, Scale data
2: **Set** $feat^0 \leftarrow D_{train}$
3: **for all** Auto-Encoder hidden (AEh) layers $l \in [1, \ldots, L]$ **do**
4: train AEh^l with input $feat^{l-1}$ and output $feat^{l-1}$
5: $feat^l \leftarrow$ apply $feat^{l-1}$ to AEh^l
6: **end for**
7: train $softmax$ layer with $feat^L$
8: $NeuralNet_{final} \leftarrow$ Stack (concatenate) All trained AEh^l layers
9: train $NeuralNet_{final}$ with $feat^0$ ▷ Fine Tuning

Fine Tuning

After separately training all the hidden sparse auto-encoder layers and the final Softmax layer, as described above, we need to stack them up. All of the trained layers are concatenated according to their training sequence chronologically. This will form the deep stacked sparse auto-encoder neural network, as shown at the lower part of Figure 5.3. The performance of this type of deep neural network can be further enhanced by training the whole network together one last time. We call this part the Fine-Tuning stage.

Table 5.2 presents the hyper-parameters that are chosen in the implementation of the Stacked Auto-Encoder. The final softmax layer used a scaled conjugate

gradient method for training and employs cross-entropy function for performance calculation. The total number of training examples is 10500, where this figure is divided into two datasets: a Training dataset comprising 80% of the samples, and a 20% Testing dataset.

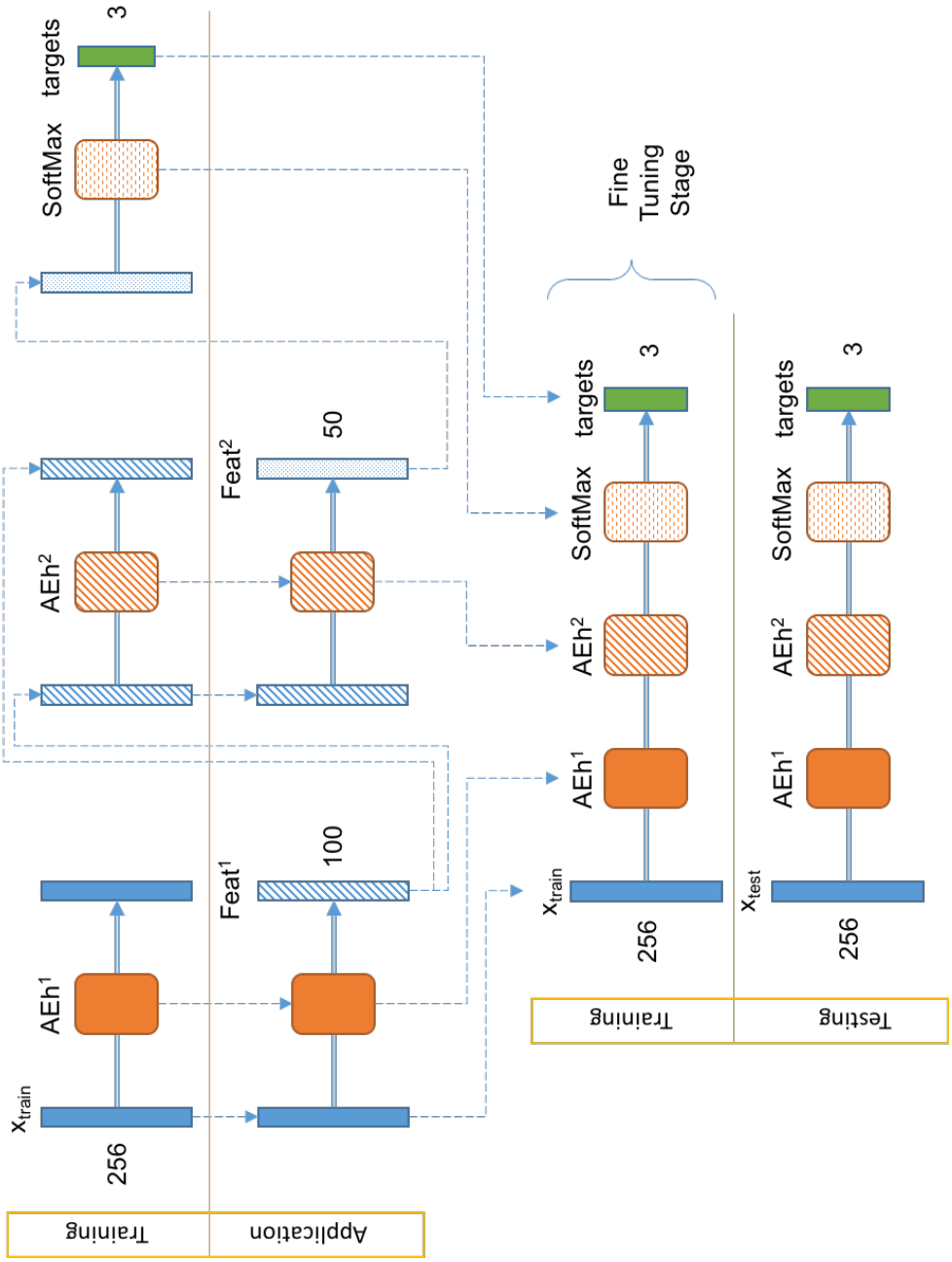


Fig. 5.3 Stacked Auto-Encoder Training and Testing Procedures

Table 5.2 Stacked Sparse Auto-Encoder Training Parameters

Parameters	Hidden Layer 1	Hidden Layer 2
Neuron (per layer)	100	50
Desired Mean Activation ρ	0.15	0.1
L^2 Weight Regularization λ	0.004	0.002
Sparsity Regularization β	4	4
Max Epochs (Layers)	500	500
Encoder Function	logsig	logsig
Decoder Function	logsig	logsig
Test/Train Ratio	1/5	1/5

Finally, in order to group classes of similar protection priorities (i.e. Primary or Secondary system), we have included a simple Mapping Function as shown in Figure 5.4. The mapper associates each class k group to a particular output as discussed previously in Table 5.1.

Stacked Auto-Encoder RAT Classification Performance

Similar to the classification performance stress test mentioned in Chapters 3 and 4, the trained Stacked Auto-Encoder is tested under a range of noisy versions of the received signal from the lab experiment described in Chapter 2. Below we describe the performance evaluation in terms of boxplots[2] figure. Here, we note that we have outperformed our previous results in Chapter 3 and Chapter 4. Figure 5.5 presents the average percentage correct classification as the SNR decreases over 30 simulations. We notice that even at SNR of -25dB and -30dB we can correctly recognize signals above 95% and 80% of the

[2]A brief description of the anatomy of a boxplot can be found in Appendix B: Boxplot Anatomy.

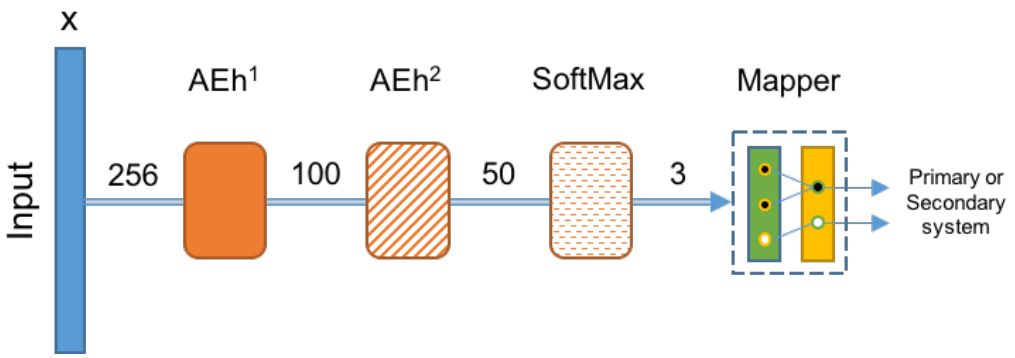


Fig. 5.4 Stacked Auto-Encoder Neural Net with Mapper Function

time (on average), respectively. These are better results than the MLP NN and SOFM-SVM architectures presented in Chapters 3 and 4, respectively. In Figure 5.6, we show another SAE architecture with an additional hidden auto-encoder layer, though we find out that it does not perform as well as results gotten from 3 hidden layers of SAE. In addition, for the sake of clarity, the internal representation of the learned features at the first hidden layer of the SAE architecture is show in Figure 5.5. Note than some of the subplots show clear resemblance of some of the features found in the raw dataset already shown in Figure 4.2.

In the next section we will show how the learned features here, namely, for a signal being classified as a *Primary* or *Secondary*, will be used in the linear functional approximator of a Q–Learning algorithm.

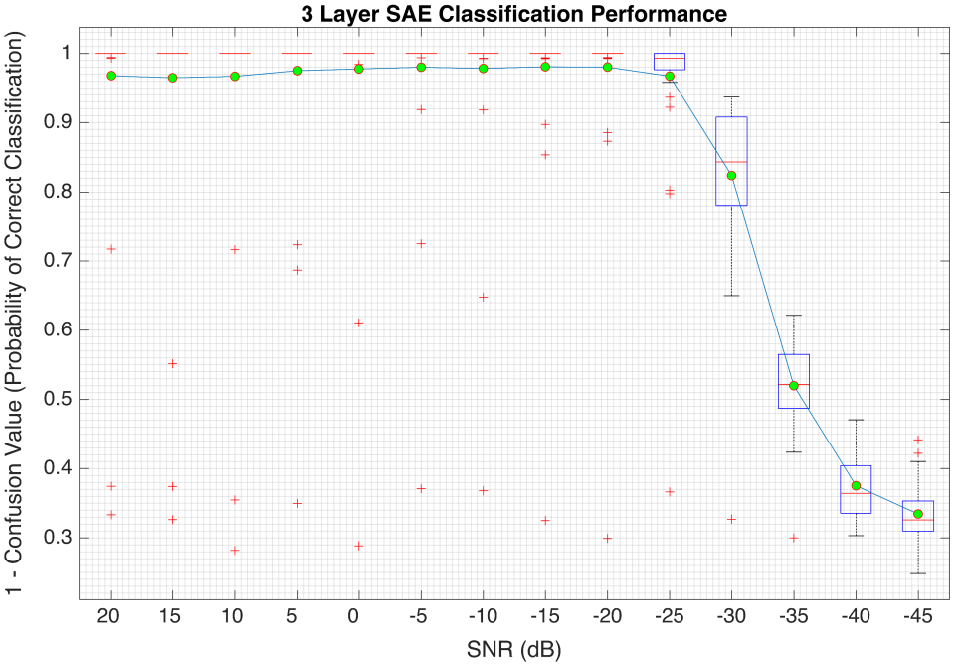


Fig. 5.5 3 Layer Stacked Auto-Encoder

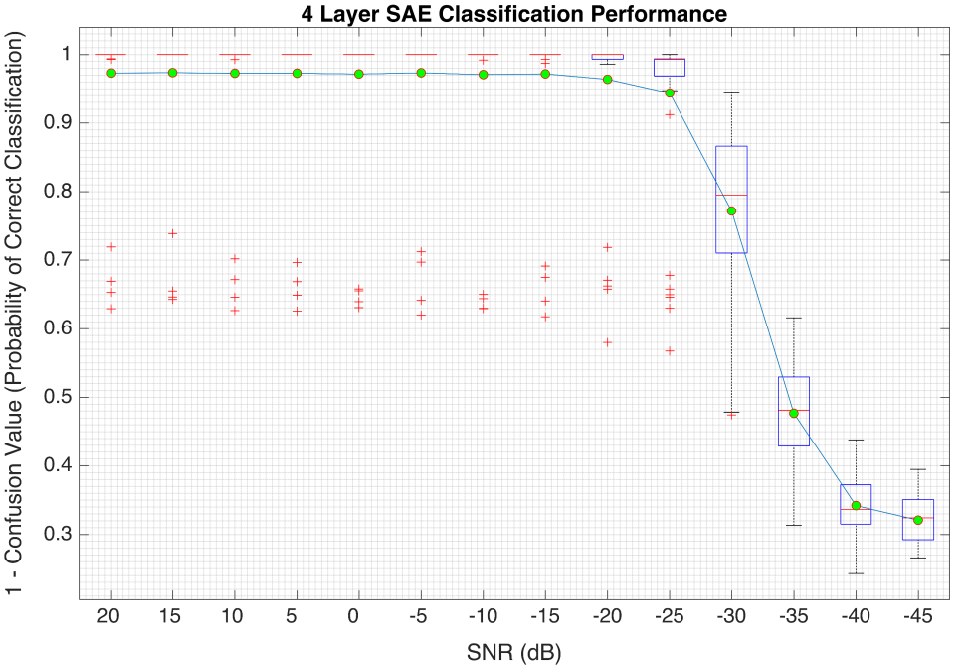


Fig. 5.6 4 Layer Stacked Auto-Encoder

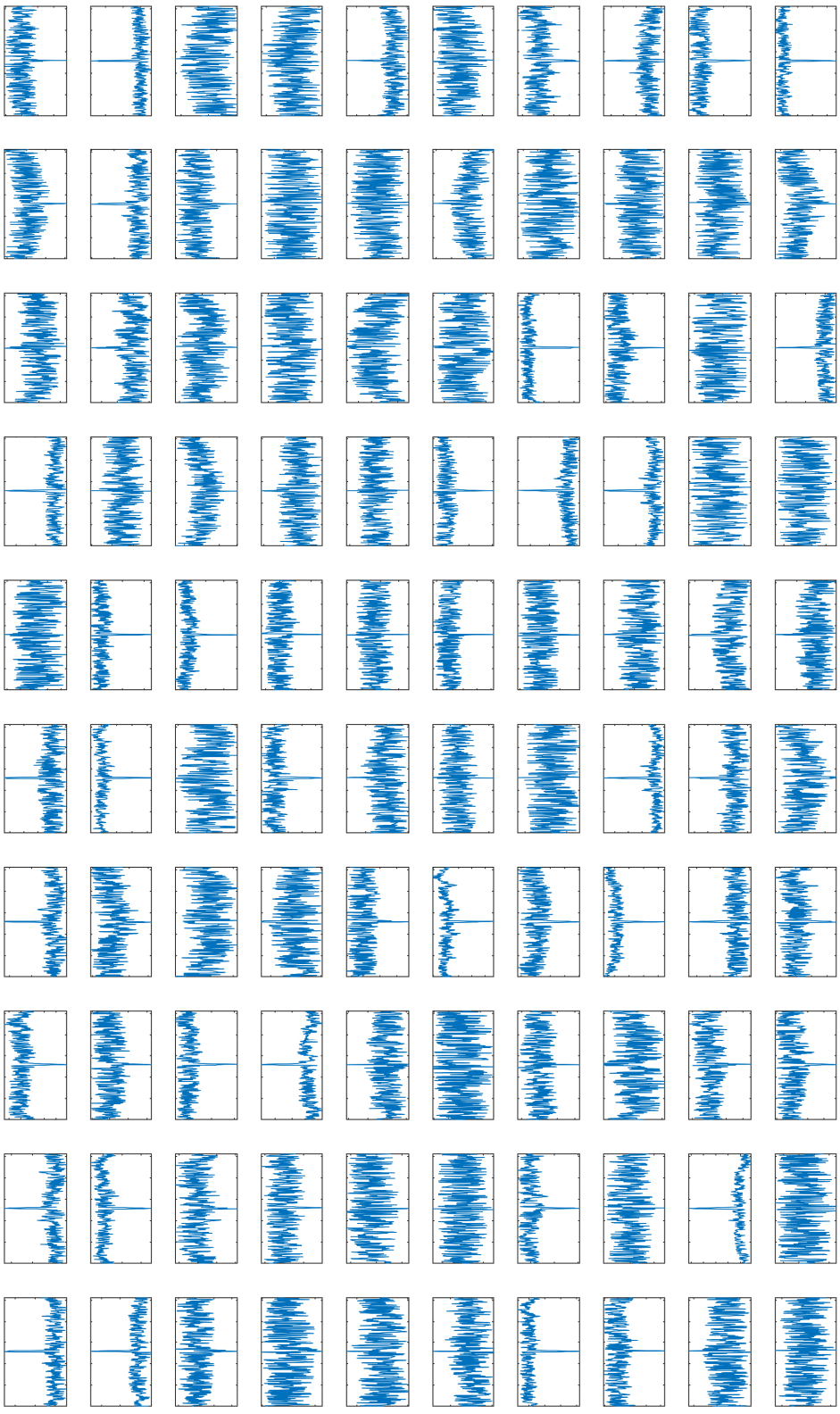


Fig. 5.7 Feature Visualization of the First Hidden Units of the 3 Layer Stacked SAE

5.4 Reinforcement Learning

In this section, we will talk about the second main point of this chapter: the reinforcement learning engine. Previously, we discussed how the RAT class was identified through the deep learning network. Here, we will use the extracted features in combination with a type of reinforcement learning algorithm called Q–Learning [176], that will decide which channel to sense and access at every time step (iteration) of the algorithm.

5.4.1 Reinforcement Learning Model

In its basic form, Reinforcement Learning problem, similar to Markov Decision Process (MDP), can be described according to the following building blocks:

1. A set of states $s \in \mathcal{S}$
2. A set of actions (per state) $\mathcal{A}$
3. A state transition model $\mathcal{T}(s, a, s')$, where next state,

$$p(s'|s, a) = \mathbf{Pr}\{S_{t+1} = s'|S_t = s, A_t = a\} \tag{5.8}$$

4. A reward function $\mathcal{R}(s, a, s')$, where next reward,

$$r(s'|s, a) = \mathbb{E}\{R_{t+1}|S_t = s, A_t = a, S_{t+1} = s'\} \tag{5.9}$$

A summary description of all the parameters used in this section is given in Table 5.3.

The type of problem

We consider a CR channel assignment task that is classified as stochastic, one-player game with incomplete information. First of all, it is a stochastic (stationary) setup, since we have modeled the primary system's packet arrival as a Poisson process (which is stochastic stationary). While there might be several radio agents in a simulation scenario, the task researched here is one-player task since we consider channel assignment problem from the point of view of one CR agent that interacts with the radio environment (except in Section 5.5.7 where there is interaction between the CRs). Furthermore, the CR learns about the channel availability without knowing the model of the radio environment; in other words, the CR does not know beforehand which channels are going to be available in the next time steps, and thus it is a *Model-Free* task, as it is called in RL literature [170].

Table 5.3 Reinforcement Learning related parameters

s, s', $\mathcal{S}$	Current state, Next state, State Space
a, a', $\mathcal{A}$	Current action, Next action, Action Space
Q	State-Action pair Q-Table/Matrix
F	State-Action Frequency F-Matrix
$\mathcal{T}$	Transition Matrix
r, $\mathcal{R}$	Reward, Reward Matrix
Pr	Probability Distribution Function
$\mathbb{E}$	Expected Value
γ	Discount Rate
α	Learning Rate
π, π^*	Policy, Optimal policy
w	Weights
f	Feature function

Reinforcement Learning tasks can in general be divided into two main categories: *on-policy* and *off-policy*. Policy-based RL, attempts at finding the optimal policy $\pi^*(s) \rightarrow a$ to achieve maximum future reward. While off-policy RL try to reach optimal value function $Q^*(s, a)$ which is the maximum achievable value under any policy [134, 135, 170].

Without dwelling too much on the background review of RL, which is a broad subject on its own, below we discuss several types of off-policy learning. We choose the well-known algorithm called *Q-learning* [176] and discuss one of its variants, called *Approximate Q-learning*, that forms the basis of the research in rest of this chapter.

System Model

We assume Heterogeneous Network (HetNet) of homogeneous ad–hoc connections between CRs which are within each other's decodable ranges. There are N CRs and a set of L uni–cast connections, $\mathcal{L}$, where $l \in \mathcal{L}$ is a single uni–cast connection. These connections, can be either primary or secondary as was laid out previously in table 5.1. Furthermore, let $\mathcal{B}_{i,t} \in \mathcal{B}$ be the set of TV channels available to a CR, $i \in \mathcal{N}$, at a particular time step, $t \in [0, T]$, where $\mathcal{N}$ is the set of CRs in the CRN and T is the End Time of the mobile CR trajectory and hence the simulation. $\mathcal{B} = \cup_{i \in \mathcal{N}} \mathcal{B}_{i,t}$ is thus the set of all TV bands considered in the CRN, with $\mathcal{B}_{i,j,t} = \mathcal{B}_{i,t} \cap \mathcal{B}_{j,t}$ representing the set of TV bands available at time t at both CR nodes i and j. We also assume the presence of common control signaling [113] between the CRs.

5.4.2 Q-Learning

Q–Learning (QL) builds a table, called Q–Table, from learning the optimal state–action pair values, called Q–values, during the training period. This table is consequently kept for reference when an action needs to be chosen given a particular state. The Q–values of state–action pairs are updated as in Equation 5.10 when $\mathcal{T}(s, a, s')$ and $\mathcal{R}(s, a, s')$ matrices are available and the system is trained in *Offline* mode, such that,

$$Q(s, a) \leftarrow \sum_{s'} \mathcal{T}(s, a, s') \left[\mathcal{R}(s, a, s') + \gamma \max_{a'} Q(s', a')\right]. \tag{5.10}$$

However, in Q–Learning, neither the transition model nor the reward functions are known to us when we first start training a model. The Q–values of state–action pairs are updated as in equation 5.11 when $\mathcal{T}(s, a, s')$ and $\mathcal{R}(s, a, s')$ Matrices are not available and the system is trained in *Online* mode.

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a')\right]. \tag{5.11}$$

Considering an MDP with several thousands of states and/or actions, the problem becomes intractable as the state-action space explodes; On the one hand, the increase in the number of states and actions increases the complexity of the problem in terms of finding the optimal policy $\pi^* = \arg\max_{a \in \mathcal{A}} Q^*(s, a), \forall s \in \mathcal{S}$, on the other hand, it will take longer to reach a certain rate of convergence.

In this chapter we propose applying an Online Approximate Q–Learning technique to track and access the TVWS spectrum band. This band is modeled similar to the actual TV channels in the United Kingdom, where each channel having a bandwidth of 8 MHz using DVB–T type transmission [4].

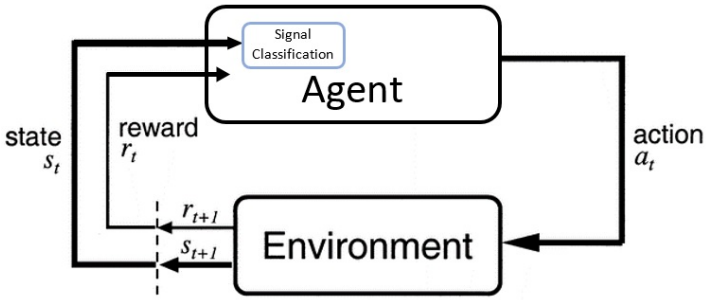


Fig. 5.8 Reinforcement Learning Cycle

5.4.3 Greedy Q–Learning

A Q-learning algorithm with greedy exploration policy as described above in Equation 5.11, is the simplest form of QL. Here, the radio agent tries to maximize its immediate reward by choosing the action that yields maximum value. It has no exploration mechanism, and thus in the channel assignment context it does not spend time exploring channels not visited. As we will see in the performance analysis section, this approach is usually inferior in performance when used in non-stationary scenarios, such as the channel access problem considered in this chapter. The Greedy QL procedure is presented in Algorithm 3.

Algorithm 3 Q–Learning Algorithm, with greedy policy

Require: α, γ

1: Initialize Q-table, arbitrarily
2: Sense the targeted TVWS spectrum band
3: **repeat** for iteration step k
4: Choose a_k such that $\arg\max_{a'} Q(s_{k+1}, a')$
5: Execute action a_k, observe r_k and s_{k+1}
6: Calculate
$$Q(s_k, a_k) = Q(s_k, a_k) + \alpha \left[r_k + \gamma \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k) \right]$$
7: $s \leftarrow s_k$
8: **until** end of episode

5.4.4 ϵ-greedy Q–Learning

A simple but powerful alternative to the greedy policy approach discussed above is the ϵ-greedy method. An agent following this policy would still try to exploit to find the channel that results in lower number of collisions, however, every now and then it would explore other channels at random with probability ϵ — hence the name [134]. Algorithm 4 displays the steps taken in ϵ-greedy Q–Learning.

Algorithm 4 Q–Learning Algorithm, with ϵ-greedy policy

Require: α, γ, ϵ

1: Initialize Q-table and $F(s, a)$ arbitrarily
2: **repeat** for iteration step k
3: Choose a_k, using ϵ–greedy policy π

$$a_k = \begin{cases} \arg\max_a Q(s_k, a) & \text{w.p. } (1 - \epsilon_k) \\ random & \text{w.p. } \epsilon_k \end{cases}$$

4: Execute action a_k, observe r_k and s_{k+1}
5: Calculate

$$Q(s_k, a_k) = Q(s_k, a_k) + \alpha \left[r_k + \gamma \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k) \right]$$

6: $s \leftarrow s_k$
7: **until** end of episode

5.4.5 Functional Q–Learning

The proposed Algorithm 5 shown below, is very similar to ϵ-greedy QL but with a subtle difference. Here, when choosing action a_k with probability $1-\epsilon_k$, rather than taking the *max* of the Q-table, we apply a *function* to the Q-table beforehand, such that,

$$a_k = \arg\max_a Q(s_k, a) + \frac{k}{F(s_k, a)} \qquad w.p. \quad 1 - \epsilon_k, \tag{5.12}$$

Algorithm 5 Functional Q–Learning Algorithm, with ϵ–greedy policy

Require: $\alpha, \gamma, \epsilon, k$

1: Initialize Q-table, arbitrarily
2: **repeat** for iteration step k
3: Choose a_k, using ϵ–greedy policy π

$$a_k = \begin{cases} \arg\max_a Q(s_k, a) + \dfrac{k}{F(s_k, a)} & \text{w.p. } (1 - \epsilon_k) \\ random & \text{w.p. } \epsilon_k \end{cases}$$

4: Execute action a_k, observe r_k and s_{k+1}
5: Calculate
$Q(s_k, a_k) = Q(s_k, a_k) + \alpha \left[r_k + \gamma \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k) \right]$
6: $s \leftarrow s_k$
7: **until** end of episode

where $F(s, a) \in \mathbb{N}^{N_{Ch} \times N_{Ch}}$ is the State-Action Frequency Matrix, and N_{Ch} is the total number of channels.

The rationale behind what we refer to the *functional Q-learning* algorithm is to extend the basic ϵ-greedy QL to be more aware of the frequency of choosing each state-action pair, such that less explored actions get higher ratings and eventually get selected.

5.4.6 Approximate Q–Learning (AppQL)

As discussed, basic Q–Learning technique builds a Q–Table which will contain the Q–Values for each state–action pair. However, when the number of the states and or actions become too large, or infinitely large, as in the case of a continuous action space [134], such as power control problem, this approach becomes intractable. Similarly, when the complexity of the problem grows, time to convergence may potentially increase as well, and we may run into out–of–memory problems. Ultimately, We would like to generalize in the state-action value function that appears in Equation 5.10.

One way to implement generalization in QL is through the use of Function Approximators [177]. Concretely, we can implement a $\widehat{Q}$–Function, which is the weighted sum of a set of manually selected feature functions,

$$\widehat{Q}(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + ... + w_n f_n(s,a), \tag{5.13}$$

where f_i is the i^{th} feature function capturing a characteristic of the input example, and w_i is the weight for feature $i \in [1,n]$.

In *regular* Q–Learning algorithm, after each transition (s,a,r,s'), we calculate the temporal difference δ,

$$\delta = \Big[r + \gamma \max_{a'} Q(s',a')\Big] - Q(s,a), \tag{5.14}$$

and then update the Q–table,

$$Q(s,a) = Q(s,a) + \alpha\delta. \tag{5.15}$$

However, in the Approximate Q-Learning Algorithm, instead, we update the weights of the Q-Function in Equation 5.16 as follows,

$$w_i = w_i + \alpha\delta f_i(s,a). \tag{5.16}$$

The pseudocode of Approximate Q–Learning approach is shown in Algorithm 6.

For the sake of clarity on how the AppQL algorithm functions, perhaps it is best to explain it via an illustration of single simulation run. Figure 5.9 below, presents (from top to bottom of the figure) feature weights w, error in terms

Algorithm 6 Approximate Q–Learning Algorithm, with ϵ–greedy policy

Require: α, γ, ϵ

Initialize all weights, W, arbitrarily
repeat for each iteration k
    Sense the targeted TVWS spectrum band
    Choose the most underutilized channel, s_k
    for all $a \in \mathcal{A}(s)$ **do**
        $f_a \leftarrow$ set of features present in s_k, a_k
        $Q_a \leftarrow \sum_{i \in f_a} w_i$
    end for
    Choose a_k, using ϵ–greedy policy π

$$a_k = \begin{cases} \arg\max_a Q(s_k, a) & \text{w.p. } (1 - \epsilon_k) \\ random & \text{w.p. } \epsilon_k \end{cases}$$

    Execute action a_k, observe r_k and s_{k+1}
    Calculate $\delta_k = r_{k+1} + \gamma \max_{a'} Q(s_{k+1}, a') - Q(s_k, a_k)$
    for all $w \in W$ **do**
        $w_i \leftarrow w_i + \alpha \delta_k f_i(s, a)$ ▷ Update W
    end for
until No more episodes

of collisions, reward gained r, and finally successful transmissions (blue circle) and collisions (red asterisk) in a 10 channel setting for the entire 1000 iteration steps. There are a few things to observe here. First, we can easily point out that for the first 100 iterations the agent is suffering from several collisions as compared to the rest of the iterations. We refer to this period as the *Learning Stage*, as depicted in the red box. Second, the agent is actively exploring more than one channel, as can be provably shown in the bottom figure. Third, the feature weights values as shown in the top figure, display precedence of used feature functions compared to each other. For instance, the feature that indicates the RAT type of the signals is ranked the highest among the others.

Having covered the necessary foundations of Q–Learning, in the next section we will conduct a thorough analyses of the Approximate Q–Learning algorithm in particular and compare it to the other algorithms discussed in this section.

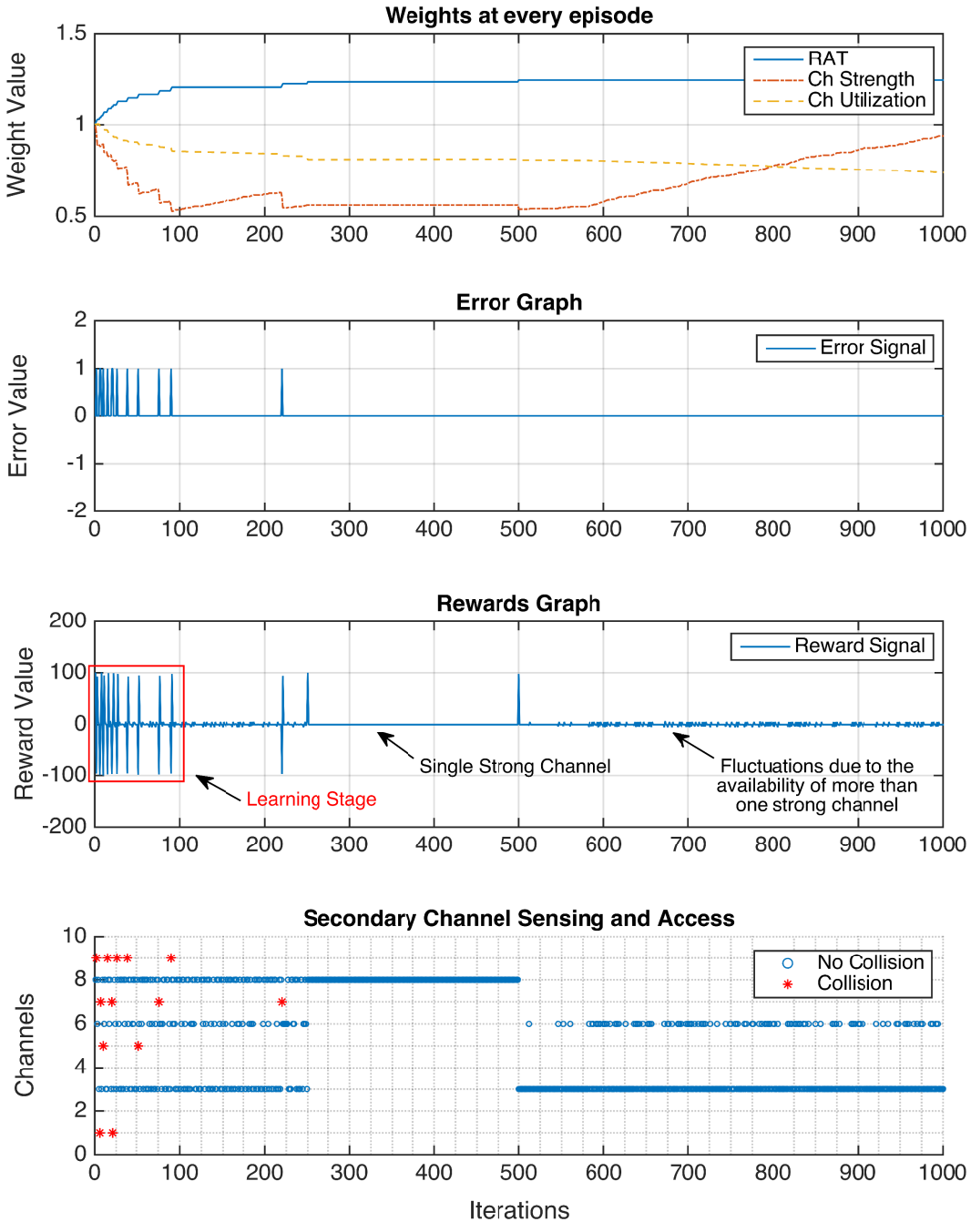


Fig. 5.9 CR using Q-Learning with Functional Approximation

5.5 Performance Analyses

In this section we put all Q–Learning algorithms discussed above to various tests. Through extensive simulations, greedy, ϵ-greedy and functional-form QL approaches are compared against AppQL over key networking related performance measures. As the chapter title also states, the main performance metric taken into consideration in this work is the *collision rate* of a secondary CR with the transmissions of a primary system. To this end, the forthcoming subsections will involve answering the following questions:

- Scalability: How does the primary-secondary collision rate change as the number of channels increase?
- Agility: How does the collision rate change when the channel jumping frequency of the primary user increases? In other words, what is the effect of an increasingly more frenetic primary user on a secondary's spectrum access performance?
- Mobility: What happens when the secondary CR is mobile? does the channel quality affect the collision rate?
- Convergence: Does the CR approach an globally or locally optimal stationary value?
- Diversity: Does the CR make use of all the channels available to it, or does it stick to one channel and hope that the receiver will not experience frequency selective fading?
- Complexity: How much time and memory does the channel access scheme take up?
- Learnability: Did the cognitive radio learn? Can its learned knowledge be useful for new agents to achieve more efficient training?

Grid–Search Optimization

In order to make a fair comparison between the different Q–Learning techniques discussed in this chapter, the hyperparameters of each method should be optimized. Hyperparameters that were optimized in this work are the following: the learning rate α, the discount factor γ, the exploration vs exploitation rate ϵ, and the special hyperparameter of the functional form Q-Learning method k.

The optimization process that was used in this work is known as Grid-Search. Each hyperparameter was optimized over a range and size as indicated by Table 5.4. Note that the table does not give the full range of values for some parameters; this is just to accelerate the lengthy simulation time. For instance, we clipped the upper limit of the parameter α so that it ends on 0.01 rather than 1; Due to our initial simulation runs, we found than all QL algorithms did better when the $\alpha \leq 0.5$. For each unique combination of these parameter values, a range of simulations were run, and the average of each simulation was stored. Later the combination of parameters that performed best were chosen. This final set was used as the optimum values to simulate each corresponding Q–Learning method and generate the figure shown therein for fair comparison purposes. In other words, each method used its own set of hyperparameters

Table 5.4 Grid Search Parameter Ranges

Parameter	Start	Step	End
α	0.0001	0.001	0.5
γ	0.001	0.01	1
ϵ	0.001	0.01	1
k	0.001	0.01	0.1

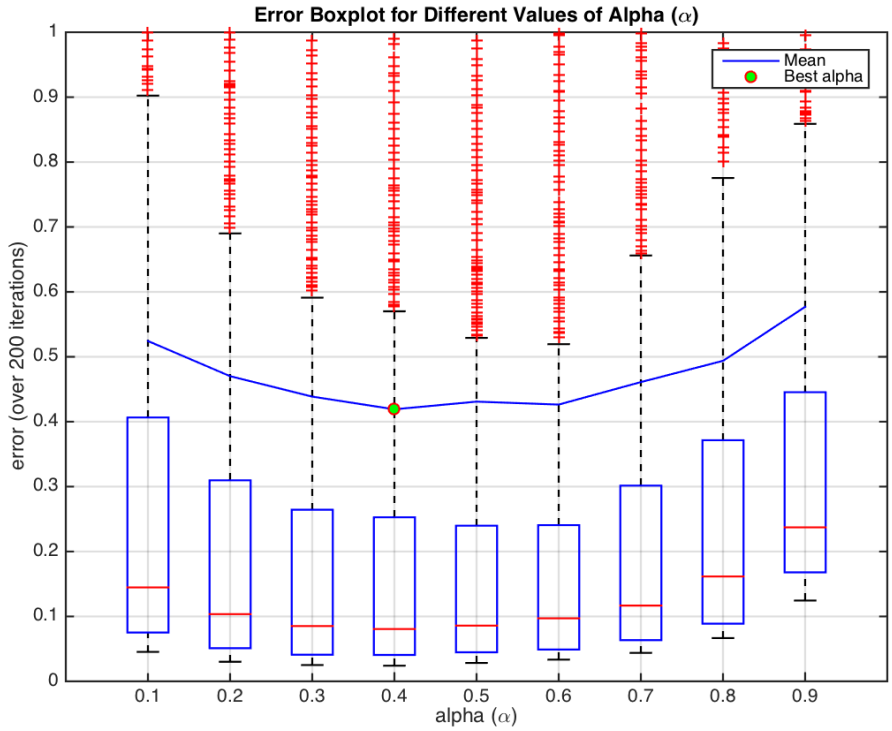


Fig. 5.10 Percentage Prediction Error Using ϵ-greedy Algorithm

rather than one common tuple of $(\alpha, \gamma, \epsilon, k)$.

For the sake of demonstration, Figure 5.10 presents an instance of optimizing one hyperparameter for one of the access mechanisms, ϵ-greedy. Here, a set of values for the learning rate α is simulated and the value that leads to the minimum amount of error (collisions) was chosen as the best α for the ϵ-greedy Q–learner.

In the subsections below, we will examine all of the performance measures mentioned at the outset of this section.

5.5.1 Scalability

Scalability is one of the key testing measures in networking-related performance evaluations. In effect, when the number of channels that are desired to be sensed and accessed are increased, an adequate channel selection algorithm's performance should not be significantly affected. In this work, by scalability testing we refer to simulating different scenarios where a cognitive radio will be sensing and attempting to access a range of finite number of channels.

For this purpose we setup 5 different sets of simulations. Figure 5.11 displays channel access resiliency in terms of primary-secondary collision rate as the number of channels are varied from 5 to 25 channels.

Here we observe that for low number of channels, e.g. 5 channels, the greedy, ϵ-greedy, and the functional form Q-Learning algorithms have similar behavior.

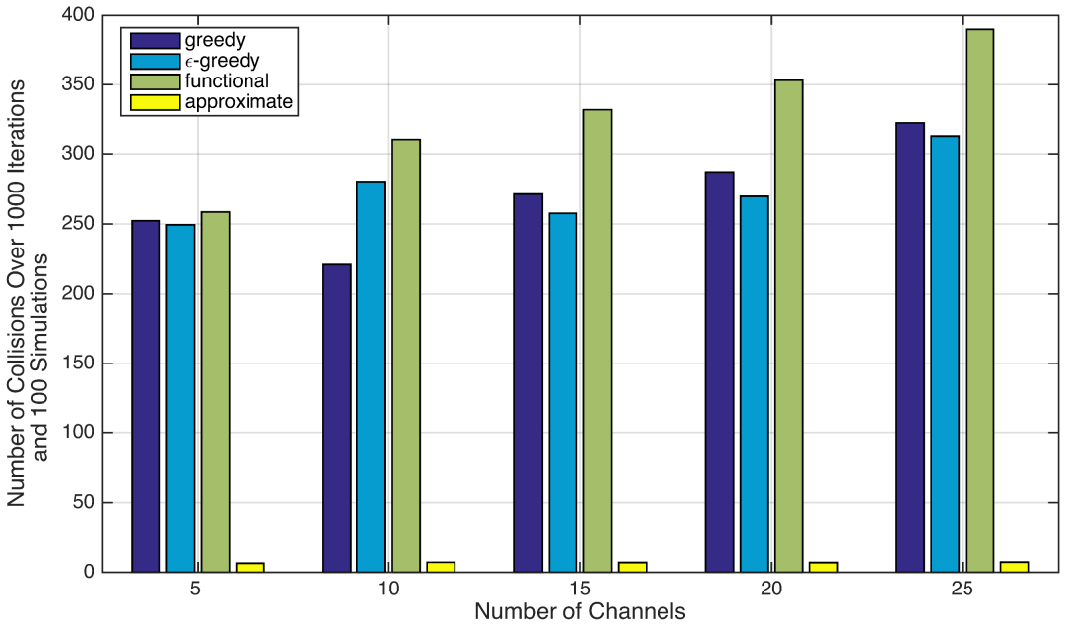


Fig. 5.11 Scalability of all algorithms as the number of channels increase, $f_d = 10$Hz, Agility=Light

However, we clearly note that these methods are severely affected when they are set to take on larger number of channels. The functional form Q-Learning exhibits the worst performance among the rest, since it takes the frequency of the state-action pair into consideration. As the number of channels (and hence actions) increase, there will be the square of number of state-action pairs in the frequency matrix. Therefore, the value $\frac{N_{Itr}}{N_{Ch}^2}$, where N_{Itr} is the number of iterations and N_{Ch} is the number of channels, will decrease exponentially. As a result, this method will be busy exploring most of the time, since it tries to uniformly populate the channel-action frequency matrix.

A more detailed figure would show finer performance evaluation of the algorithms. Specifically, considering the low number of channels cases, e.g. 5 or 10 channels, as shown in Figure 5.12, we can better note that the lower-bound whiskers (representing 25^{th} percentile) of the greedy and ϵ-greedy methods achieve the same collision rate as the 75^{th} percentile of AppQL. We suspect this to be due to the fact that the experiment was run in agility mode 4, meaning, for the whole duration of the experiment (1000 iterations) the primary user changes its channel 3 times (every 250 iteration). In other words, the primary system channel dynamics is not (relatively) very high. As a result, there might be a case where one channel will be free for the entire duration of the simulation run (1000 iterations), and thus a greedy policy should play very well in such a situation.

As for the AppQL algorithm, we note that it exhibits a stable performance throughout the test. A few collisions are recorded, and these happen during the learning stage as previous shown in Figure 5.9.

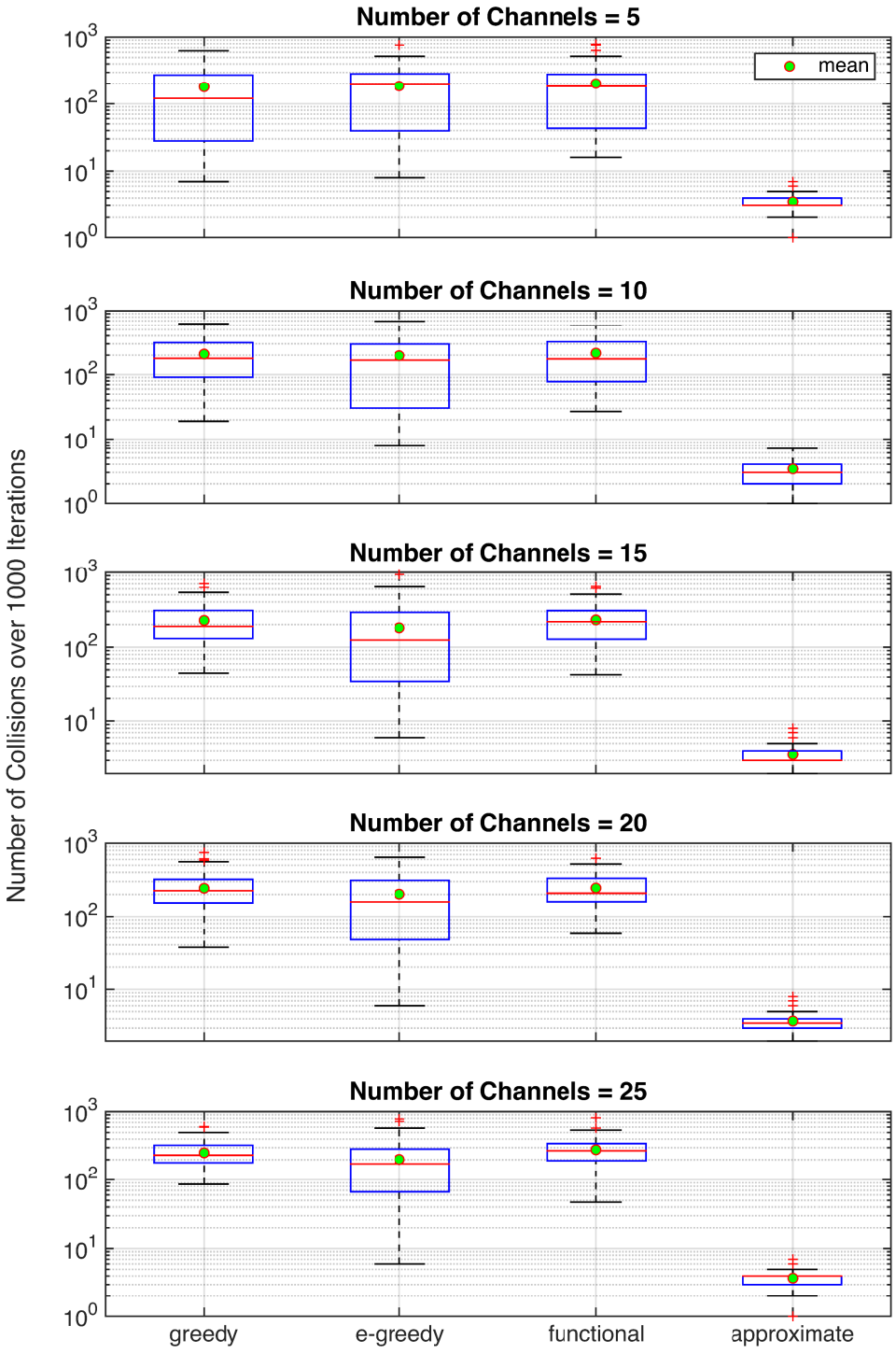


Fig. 5.12 Log-scaled Boxplot of Primary-Secondary Collisions as the Number of Channels Increase, averaged over 100 simulations, $f_d = 10$Hz, Number of PU channel jumps=3.

5.5.2 Agility

In this work, we consider Cognitive Radios to coexist alongside Primary systems who also would like to make use of the best transmission opportunities are they arise. For instance, a primary mobile user, could be informed to transmit on different TVWS channels as they traverse an area where TVWS channels are shadowed out. Therefore, we simulate the wireless traffic on the TVWS band as a Poisson arrival process with arrival rate λ as the average number of arrivals in a unit of time. The arrival probability of n packets in time interval, T, then can be modeled as,

$$P_n = \frac{(\lambda T)^n}{n!} \exp(-\lambda T) \tag{5.17}$$

and therefore the inter-arrival times are exponentially distributed, such that,

$$P(\text{interarrival time} > t) = \exp(-\lambda T). \tag{5.18}$$

Figure 5.13 displays three different realizations of a primary system's agility, namely, Dense, Moderate and Light activity.

The collision rates from the three different scenarios are shown in Figure 5.14, where the $N_{Ch} = 10$ and exponentially decrease in collisions as the primary system agility decrease linearly.

5.5.3 Mobility

A CR may be mobile as well, and thus could be relocated to other locations where the radio environment is different from where it was before. Following a trajectory, a CR shall be able to adapt to the available channels along the

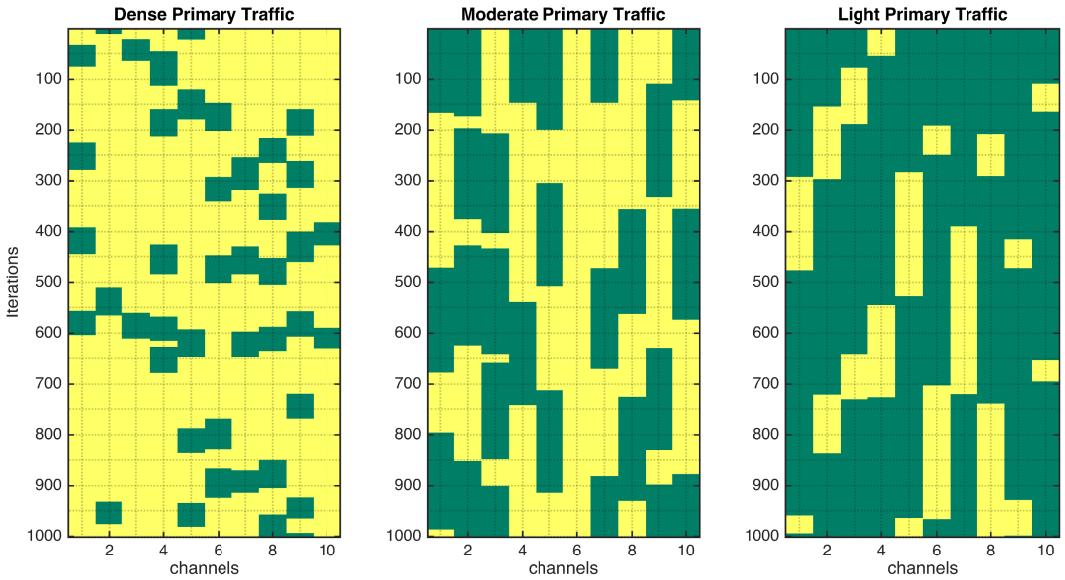


Fig. 5.13 Three Different Realizations of Primary System Agility Over 10 Channels. Yellow means the the primary user is on, and green indicates the primary signal is not identified.

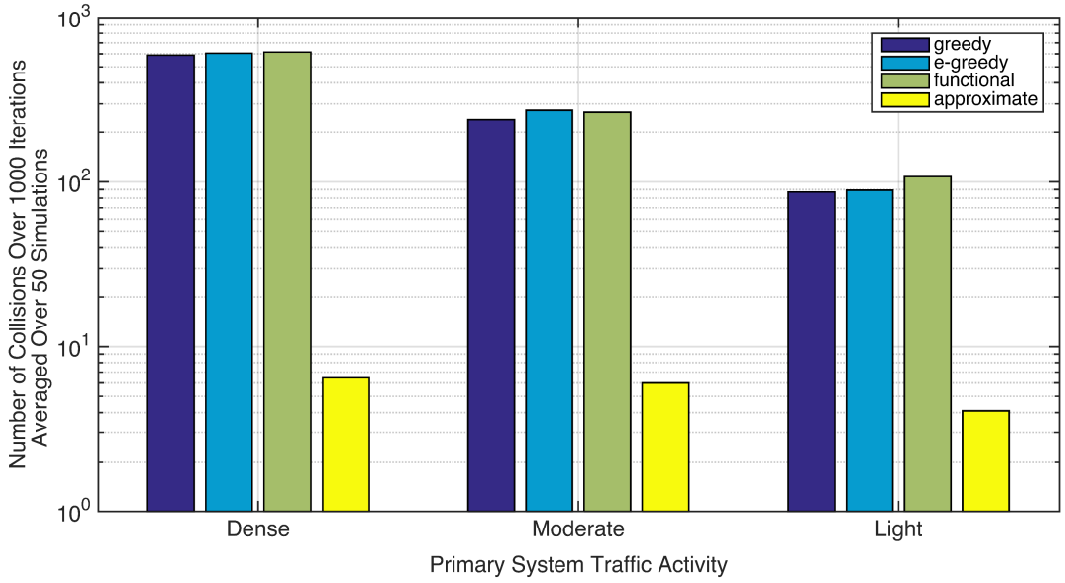


Fig. 5.14 Log-Scale Secondary-Primary Collision Rate in Different Primary System Traffic Models with Dense, Moderate and Light PU channel jumping frequency. $N_{Ch} = 10, N_{Sim.\ Runs} = 50$, Avg. Packet Size = 200

path. This adaptation could be shown through a heat map of the states, or in other words the frequency of choice of the available channels. Furthermore, TVWS is not totally sporadic but exhibit a spatial correlation. In this section, we will consider a CR, equipped with the above discussed QL learning algorithms, and traversing an geographical area while it tried to solve the channel assignment problem at the 700Mhz TVWS band. In such a scenario, our aim is to test whether channel fluctuations caused by being mobile will affect the performance on the CR.

Mobility Model

The scenarios we have considered so far were of a static CR sensing the radio environment and testing its performance to access the TV white space. Here, we will test the CR's collision rate while we mobilize the CR in simulation from point A to point B at different speeds. The set of velocities considered are: 10 km/h, 100 km/h and 360 km/h to yet again test the convergence performance of the Approximate Q–learning technique under higher mobility.

The chosen channel model in the scenario is the 3GPPTUx model (TU for Typical Urban) [178], which is a simplified realization of the COST 259 model [179]. The propagation properties that were considered in COST 259 can found in Table 5.5. Coherence Time decreases inversely proportionally with Doppler Shift, as indicated by Equation 5.19.

$$\text{Coherence Time} \approx \frac{1}{\text{Doppler Shift}} \tag{5.19}$$

As the Doppler spread is increased, the coherence time decreases and thus it will make it harder for the CR to settle in a stable channel. Table 5.6 lays

Table 5.5 Propagation Properties considered in COST 259 Model [179]

1	Path Loss
2	Shadow Fading
3	Fast Fading
4	Time Dispersion
5	Angular Dispersion
6	Polarization
7	Multiple Clusters
8	Dynamic Channel Variation

out three different instances of this model that has been taken into account in regards to stationary and vehicular modes, based on the maximum Doppler shifts calculated according to Equation 5.20 [150]. The Maximum Doppler Shift f_d is calculated such that,

$$f_d = f_c \frac{\mathrm{v}_{rel}}{c}, \tag{5.20}$$

where v_{rel} is Relative Velocity, f_c is the Carrier Frequency and c is the Speed of Light (3×10^8 m/s). An instance of the simulated channel model for the mentioned three scenarios in Table 5.6 is shown in Figure 5.15.

Table 5.6 Mobility Scenarios indicating Relative Velocity and Maximum Doppler Shift

Scenario	v_{rel}	f_d
Semi-stationary	10 km/h	7 Hz
Urban vehicular	100 km/h	65 Hz
Highway vehicular	360 km/h	234 Hz

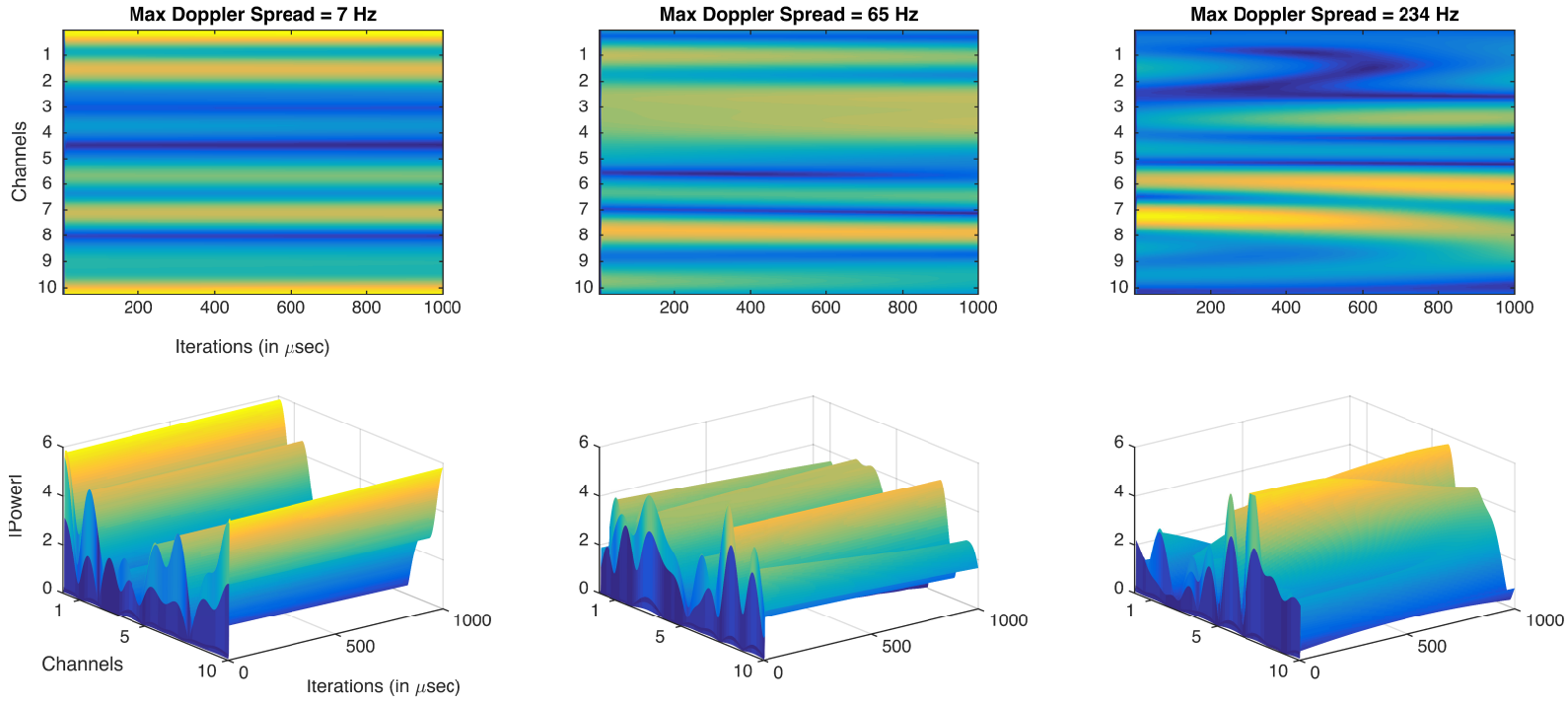


Fig. 5.15 Typical Urban 3GPPTUx based Correlated Freq-Time Channel simulation for 10 channels over 1000 iterations (≈1 msec) with three different Maximum Doppler Spreads: Semi-stationary $f_d = 7$Hz (left), Urban Vehicular $f_d = 65$Hz (middle), High-Speed Vehicular $f_d = 234$Hz (Right). Note that for the latter case the channel response is not constant throughout simulation duration.

The collision rate results regarding the above mentioned three mobility scenarios are shown in Figures 5.16, 5.17 and 5.18. As a general remark, we can note that in the light PU agility scenario all methods except the Functional Q-learning method do relatively well in terms of the number of collisions incurred. As the level of PU agility increases all methods collision rates increase, nonetheless, AppQL suffers around 6-times less than the other methods at the Dense PU agility case.

In regards to mobility, we could not notice a major effect of increasing the Maximum Doppler Shift f_d. The reason for this is mainly due to operation in a low frequency band where in this case we have run our simulations at 700 MHz band – resembling a TVWS radio environment.

5.5.4 Diversity

A general phenomena that arises in wireless communications is that on Rayleigh fading channels the attenuation in Bit-Error-Rate (BER) as the SNR increases is linear, and not exponential as it is with Additive White Gaussian Noise (AWGN) Channels [150]. Herein, a heuristic measure is proposed to quantify the degree of diversity provided by the algorithm under research.

In order to explain the intuition behind such a heuristic measure, which will be referred to as the *Channel Utilization Diversity (CUD)*, consider the three different CR channel access instances shown in Figure 5.19. In the first case (a), the CR depicted in blue, achieves maximum channel diversity since it visit all of the channels equally in the three iterations available to it. In (b), the black CR achieves sub-optimal diversity since it does not visit the middle channel while there is an opportunity to do so, has it visited the channels in a different order, e.g. (1,3,2). The final case (c), only achieves minimum diversity since it

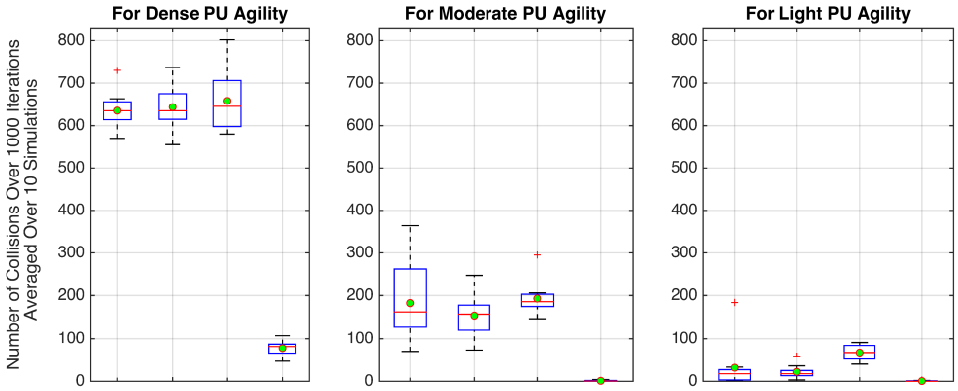


Fig. 5.16 Mobility test result at maximum Doppler frequency $f_d = 7$ Hz

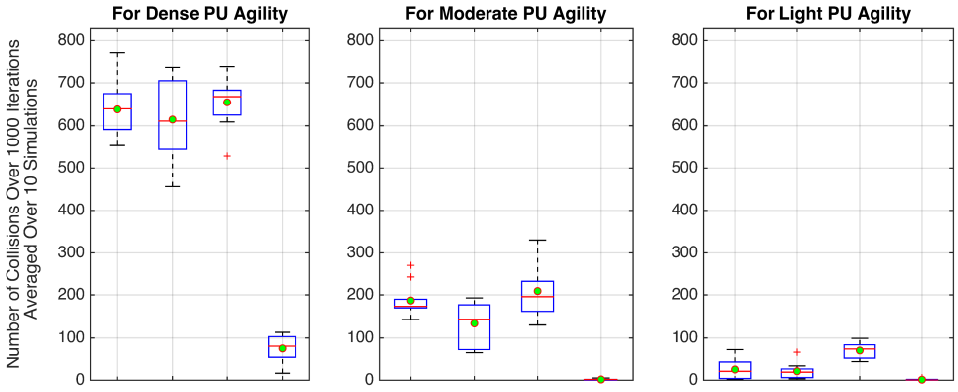


Fig. 5.17 Mobility test result at maximum Doppler frequency $f_d = 65$ Hz

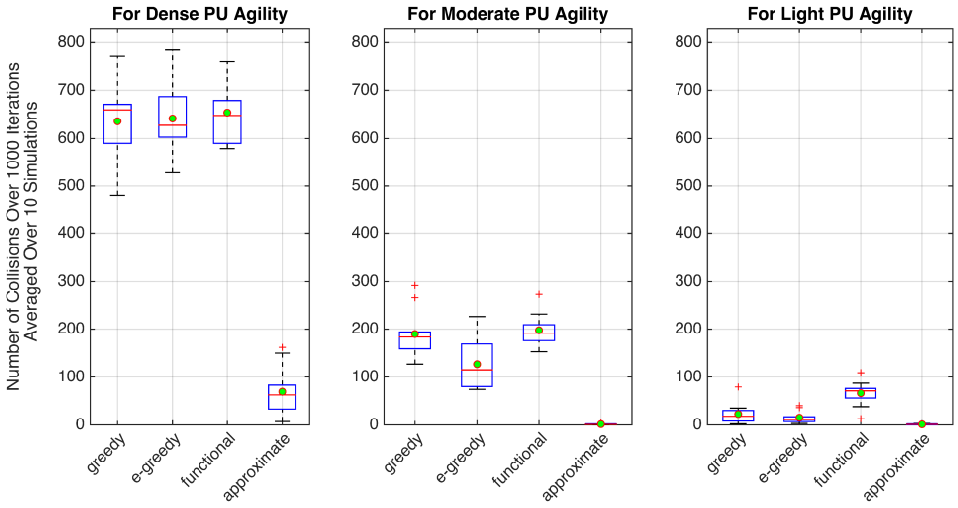


Fig. 5.18 Mobility test result at maximum Doppler frequency $f_d = 234$ Hz

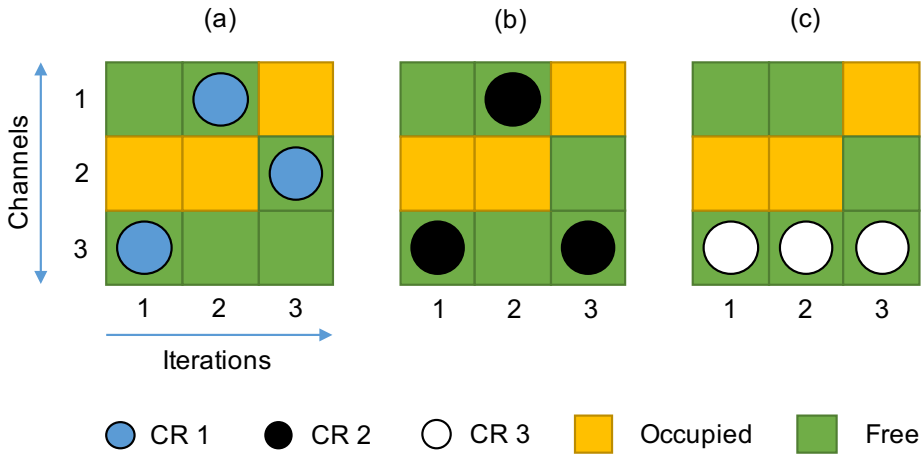


Fig. 5.19 Explaining Channel Utilization Diversity (CUD) Measure

only exploits one channel for the duration of all iterations. Thus, the heuristic measure designed should give maximum value to case (a) and minimum to (c), while (b) should be assigned a value somewhere in between.

For a fair calculation of CUD, the capacity vector $C^{div} \in \mathbb{R}^{N_{Ch}}$ of the available channels for secondary transmissions weighted by the complement of the channel strength matrix $S'_{c,i} \in \mathbb{R}^{N_{Ch} \times N_{Itr}}$ shall be taken into account, where N_{Ch} is the number of channels and N_{Itr} is the number of iterations in a simulation experiment. Here the complement of the channel strength matrix is defined as

$$S'_c = 1 - \frac{S_{c,i} - S_{c,(min)}}{S_{c,(max)} - S_{c,(min)}} \; \forall i \in [1, N_{Itr}]. \tag{5.21}$$

As an example, consider C_c^{div} for all channels $c \in [1, N_{Ch}]$ in Figure 5.19 is 2 (where $c = 1$) + 1 (where $c = 2$) + 3 (where $c = 3$) $\Rightarrow C_{div} = \{2, 1, 3\}^T$. More formally,

$$C_c^{div} = \sum_{i=1}^{N_{Itr}} \mathbb{I}(\neg \text{PU} \mid c, i) S_c, \tag{5.22}$$

where $\mathbb{I}$ is the Indicator function, and its use in the equation above implies that the output will be equal 1 if the channel c sensed, at iteration i, is not occupied

by a PU. Additionally, let the matrix $F \in \mathbb{R}^{N_{Ch} \times N_{Itr}}$ be recording the channels a CR visits at every iteration. Then CRs' transmissions in every channel for the duration of each simulation run (1000 iterations) can be presented as a vector T, such that,

$$T_c = \sum_{i=1}^{N_{Itr}} F_{c,i}, \tag{5.23}$$

where $i \in [1, N_{Itr}]$ and $T \in \mathbb{R}^{N_{Ch}}$. Finally, we define CUD to be,

$$\text{CUD} = \sum \frac{\sum -C_{div} \times (T - C_{div})}{\max(\sum -C_{div} \times (T - C_{div}))} \tag{5.24}$$

which can be put more conveniently as,

$$\text{CUD} = \sum \frac{\sum C_{div}^2 - TC_{div}}{\max(\sum C_{div}^2 - TC_{div})} \tag{5.25}$$

The results obtained from running 100 simulations of 1000 iterations each for all the algorithms according to Equation 5.25 under different PU agility measures are shown in Figure 5.20. Here it can be noticed that the AppQL and the functional form of Q-Learning show approximately similar performance. The good results that the functional form achieves here are a manifestation of its exploration property which was essentially enabled by the taking the channel frequency matrix into account. In other words, in the functional Q-Learning algorithm if a channel was less favored, e.g. visited less by a CR due to its channel strength properties, its channel frequency feature function would be higher in value and eventually making this channel more likely to be explored by the CR every now and then.

The greedy policy, as was expected, did not do well in this test due to its greedy nature to exploit rather than explore and hence achieved the worst diversity performance. On the other hand, the ϵ-greedy algorithm, since it

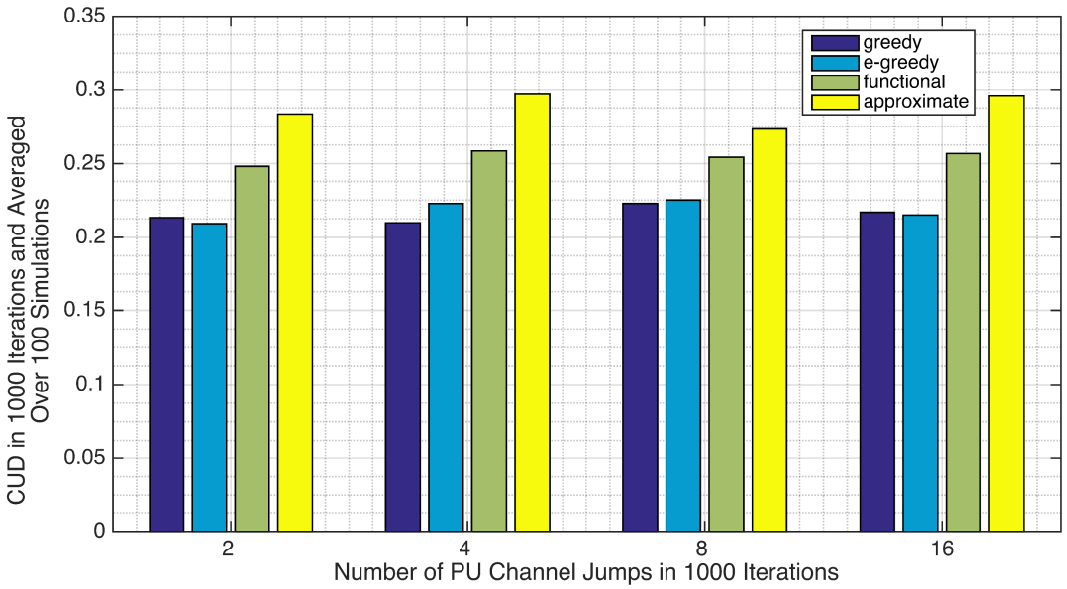


Fig. 5.20 Cognitive Radio Channel Utilization Diversity (CUD). (a) Maximum diversity (b) Intermediate diversity (c) Minimum diversity. $N_{Ch} = 25$ Channels, and PU Agility=2^a jumps, where a=1,2,3,4.

explores other channels randomly with probability ϵ, behaved better than the greedy algorithm in all the different PU agility scenarios, yet worse than the Approximate and functional Q-Learning Algorithms.

5.5.5 Complexity

Complexity is another important aspect of algorithms that needs to be evaluated. Here, we investigate the empirical time and theoretical space complexities.

Time Complexity

The time complexity in this work is only empirically evaluated; while it is common to measure it theoretically however in this case it would be fairly complicated. Essentially, here we have recorded the time taken by the implemented

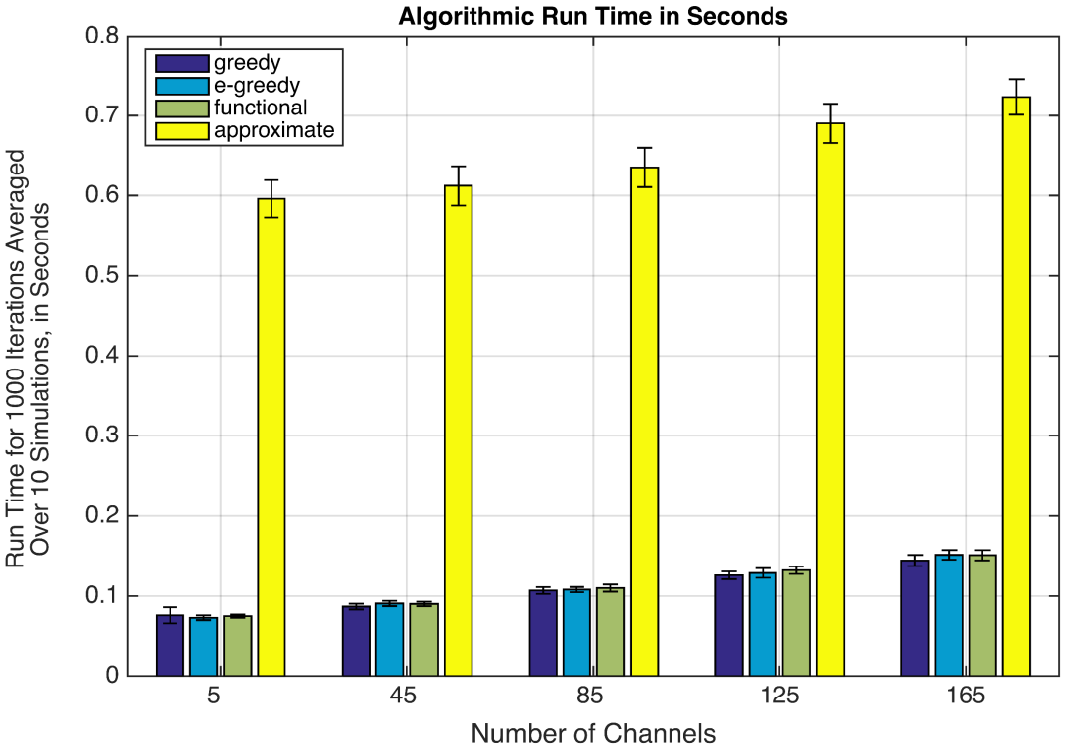


Fig. 5.21 Algorithmic Time Complexity

algorithm for every simulation episode and then repeated the experiment 10 times, taking the average over all simulation runs. The bar graph shown in Figure 5.21 indicates the time taken by each QL algorithm as the number of channels was increased from 5 to 165 in increments of 40. Due to the simplicity of the greedy-QL algorithm, it achieves the best delay performance. This is due to the fact that it does not even run on a probabilistic exploration policy as compared to the other QL algorithms. The greedy-QL rather sticks to the channel that it expects to give the maximum reward in the next time step. While this might be a sound exploitation strategy, it does not fit very well for wireless communications where the radio environment is intense and non-stationary.

On the other hand, the ϵ-greedy and functional Q-Learning methods behave similarly, though taking slightly longer than greedy-QL. However, the time

taken by the proposed AppQL is found to be several times longer than its counterparts, which can be considered a major draw-back of this technique. The reason for this is mainly due to the fact that the Approximate Q-Learning algorithm employs three feature functions that needs to be evaluated on every iteration, and thus would entail an additional time and computational costs.

Among all a performance analyses done in this chapter, time-complexity happens to be the biggest caveat of the AppQL. Nevertheless, if we take a closer look at Figure 5.15, we note that for both the $f_d = 10$ Hz and $f_d = 100$ Hz, the channel response is quite stable. In other words, the coherence time is expected to be longer than what was simulated i.e. $\tau_{\text{Coherence}} > 1$msec, and thus for such sub-GHz spectrum bands packet size could be longer. Finally, as a general remark, it is obvious that with the increase of the set of channels considered, the delay also increases. However the *rate* of time delay increase on

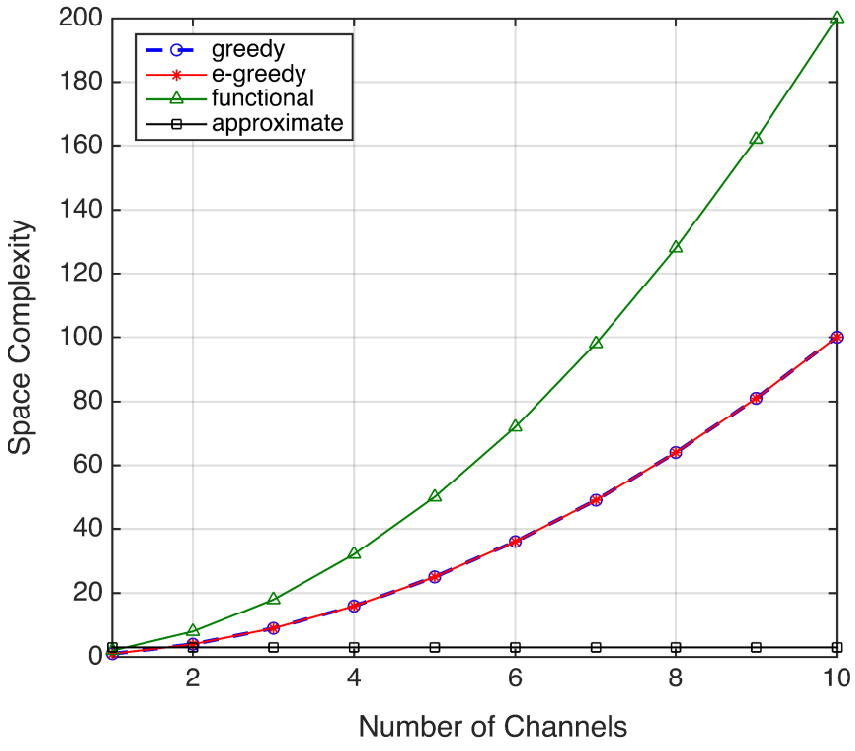


Fig. 5.22 Algorithmic Space Complexity

the AppQL is noted to be slightly more severe than the rest of the algorithms. This is most probably the direct cause of extra computation time needed to consider the *max* function inherent to the AppQL algorithm.

Space Complexity

Calculating the space complexity, or in other words, the Random Access Memory (RAM) taken by the algorithms is more straight-forward. To calculate this figure, we shall ask: what are the size of variables (in memory) that we need to store in-between subsequent iterations for the algorithm to proceed? In this case, the answer essentially is the size of matrices the algorithms store. Figure 5.22 indicates the space required by every algorithm, where an element of a matrix is counted as one unit of memory.

For the tabular greedy and ϵ-greedy Q-Learning algorithms, their Q-tables depend on N_{Ch}. As the number of channels increase, the space complexity also increases with exponent 2 (since the Q-tables are square tables). The functional-QL execution entails the storage of an additional matrix $F_{c,i}$ as was previously discussed in Equation 5.23, and thus suffers the most among the simulated QL methods. Finally, we note that the algorithmic space complexity of the AppQL stays constant, since what we essentially retain between every iteration is the learned feature function coefficients. This in effect achieves one of the main goals of function approximation techniques.

5.5.6 Convergence Test

Convergence of a reinforcement–based learning algorithm is an important property where approaching the (optimal) solution is guaranteed. However, it has been shown that Approximate Q-Learning may diverge in some scenarios. In

an off-policy setting, Q-learning with function approximation may oscillate about an optimal solution, leading to a never-ending learning process within the prescribed error margin. This has been found to be due to the fact that TD(0) does not implement true gradient, and it can be shown as follows [180]. From Equation 5.14, let $\gamma \max_{a'} Q(s', a')$ be equal to $\gamma w^T f^*(s, a)$ where $f^*(s, a)$ is vector of function values $f_{1...n}(s, a)$ evaluated at the action a that maximizes the sum of each of these function values. Thus, Equation 5.14 can now be re–written as,

$$\delta = r + \gamma w^T f^*(s, a) - w^T f(s, a). \tag{5.26}$$

Assume there is a parametrized cost function, $J(w)$, needs to be minimized via gradient descent. First, a sample gradient, $\nabla_\theta J_t(w)$, is found analytically such that

$$\frac{\partial J}{\partial w_i} = \delta f_i. \tag{5.27}$$

Then, small steps are taken in w in the direction of this sample gradient such that

$$w \leftarrow w - \alpha \nabla_\theta J_t(w). \tag{5.28}$$

However, taking the second derivation of the cost function, $J(w)$, to test for Symmetry,

$$\frac{\partial^2 J}{\partial w_j \partial w_i} = \frac{\partial(\delta f_i)}{\partial w_j} = (\gamma f'_j - f_j) f_i \tag{5.29}$$

$$\frac{\partial^2 J}{\partial w_i \partial w_j} = \frac{\partial(\delta f_j)}{\partial w_i} = (\gamma f'_j - f_i) f_i \tag{5.30}$$

this cost function is found to be asymmetric [181], in that, Equation 5.29 is not equal to Equation 5.30.

However, on the positive side, Q-Learning with linear function approximation is found to be (almost surely) converging if the follows are held true [135, 172]:

1. A linear function approximation is used $f : \mathcal{S} \rightarrow \mathbb{R}^d$
2. The stochastic process governing the state space $\mathcal{S}$ is an ergodic Markov process.
3. the learning rate α satisfies the Robins-Monro (RM) conditions,

$$\sum_{t=0}^{\infty} \alpha_t = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2 < +\infty \tag{5.31}$$

More recently, RL research has attracted significant attention especially with the emergence of faster [181] and convergent [182] off-policy algorithms. In particular, in [180] a gradient-like Temporal Difference algorithm, namely, greedy-GQ has been proposed which is not based on the hard conditions set out by [172].

AppQL Convergence Results Considering experimental results which hint at AppQL's convergence as well, Figure 5.23 present the weight variance with 2.7 standard deviation shaded confidence bounds. While Figure 5.24 is a further simplification of Figure 5.23, showing boxplots of the weights of each feature function averaged over 100 simulation runs.

Through these two Figures, we also note the importance of each feature weight value. Here we can see that the RAT feature is the most critical parameter with respect to the channel strength or channel utilization feature. This is evidently so, since the RAT feature contributes the highest reward and incurs the highest punishment (or negative reward) to the simulated reward function (as the reward function was intentionally designed to behave so). The channel strength feature comes second in this matter as it was also taking up the role to be the deciding feature when two channels of equal access priority (either both primaries or both secondaries) was encountered. Therefore, the

channel strength feature weight would also be considered important, yet not as important as the RAT feature. Finally, since channel utilization was not accounted for in the reward function (and this is rightly so since its an internal feature of the CR), its weight value was only updated when the CR thought a channel has not been explored with respect to the other channels. In this case, the contribution of the weight value of the channel utilization feature of a such a channel would be high enough to candidate the mentioned channel to be accessed and explored.

Additionally, for the sake of comparison Figure 5.25 gives the mean collision rates for all mentioned algorithms herein. Here the primary changes its frequency at certain time instants, to be adequately compared with one another. It is clearly evident that all algorithms converge in practice since their collision rates decrease after every primary users' jump. However, it is also very evident that not all methods converge at the same rate. After the first primary user jump, all of the QL algorithms' learning rates are deteriorated, as was implemented intentionally. After obtaining the initial impression of the channels availability, they become a slower learner there after. As for the AppQL, however, we notice it commits minimal mistakes throughout, thanks to having learned which features, if detected, are important and how to avoid them.

5.5.7 Learnability

A very important criteria to evaluate the level of cognition of a cognitive radio is the amount of knowledge learned in a finite period of time. In reinforcement learning, the agent learns from trial and error which results in positive rewards or negative feedback on its interaction with the surrounding environment, as was shown in Figure 5.8. In this sense, the learning process is sequential

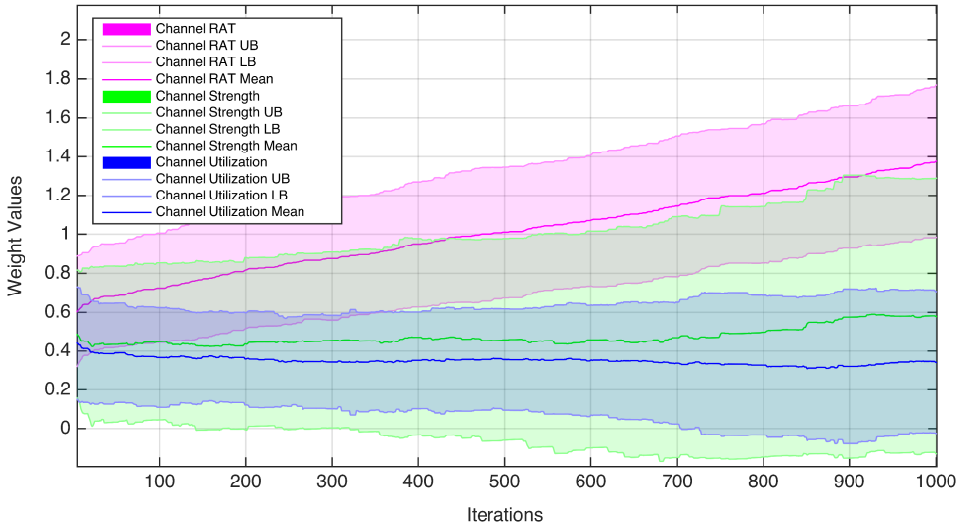


Fig. 5.23 Approximate Q–Learning Empirical Weight Convergence

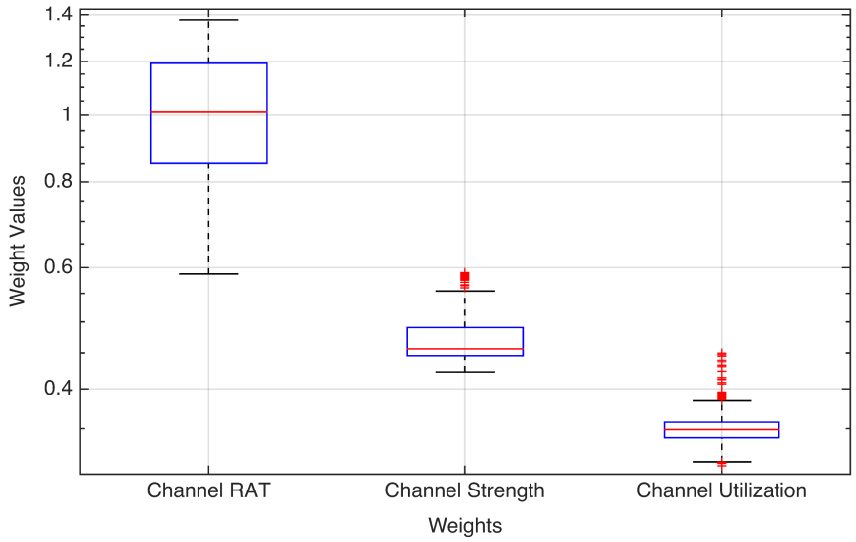


Fig. 5.24 Box Plots of the Weight Coefficients (logarithmic scale) Used in Approximate Q–Learning Algorithm

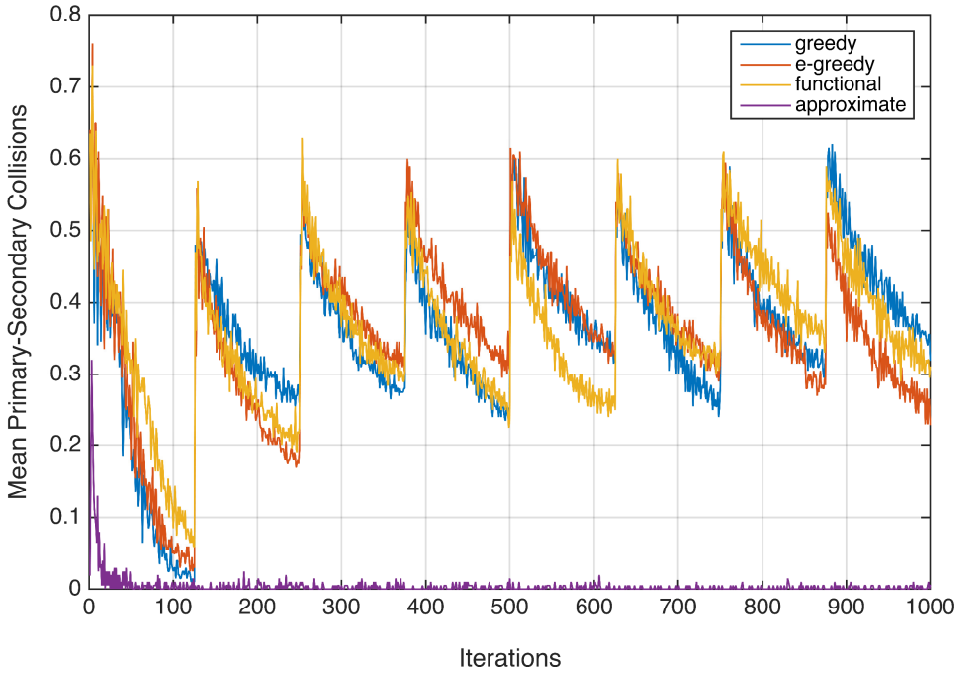


Fig. 5.25 Collision Rate for all Algorithms with 7 Primary Channel Jumps

in nature, based on a series of actions taken. In a complex learning problem, however, this learning process is indeed time as well as energy consuming [183]. Researchers have tackled this problem in Machine Learning literature in the subject of Transfer Learning (TL) [184, 185]. There is a substantial amount of literature on transfer learning, especially recently because of success of RL in various applications as discussed in Section 5.1. In the context of cognitive radio, transfer learning has also been applied in [186–188] to balance Quality of Service (QoS) and information exchange overhead in channel assignment task with up to 50% [186] reduction in retransmissions. Likewise, TL has been used in green communications context, where in [189] information is exchanged between radio agents to achieve more efficient basestation on-off switching. Authors in [76, 190, 190] have coined the term "Docitive Networks", where they refer to mature radio agents teaching less experienced agents.

To elaborate further, perhaps is it worth identifying the difference between *information transfer* and *knowledge transfer*. According to Oxford dictionary, *Information* in computing is defined as, "Data as processed, stored, or transmitted by a computer". *Knowledge* on the other hand, is defined as "Facts, information, and skills acquired through experience or education". In this context, co-located cognitive radios that share the same radio environment, can exchange *information* such as TVWS channel availability (if co-located close enough) or channel RAT types, at a *specific time and location*; while, if it was desired to share *knowledge*, this could be in terms of the types of features the CR uses or the ranking of the features importance, e.g. RAT feature is more important than average channel PSD (as discussed in section), which is true at different times and places. Therefore, here, by *knowledge* we refer to information that can be generalized upon. In this sense, knowledge data would be still valid (in a specific context) regardless of spatial or temporal differences.

Putting the above into practice, it will be essential to find out whether the RL-based radio agents discussed in this chapter actually gained knowledge after training, such that previously learned knowledge can be harvested and disseminated to reduce the above mentioned costs in any future learning, be it for the cognitive radio itself, or to transfer the learned knowledge to another agent radio. Putting our cognitive radios to test, let us define a 3 stage scenario as shown in Figure 5.26. In stage one, Figure 5.26a, CR_1 using AppQL algorithm is learning the channel selection task. In stage 2, if CR_1 improved on its channel access performance e.g. through increased reward or reduced collisions, CR_1 then transfers its feature weight values w (see Equation 5.13) to CR_2. In stage 3, again, if CR_2 improved on its channel access performance, CR_2 then transfers its feature weight values w to CR_3.

As shown in Figure 5.27, the taught cognitive radios, i.e. CR_2 and CR_3, startup

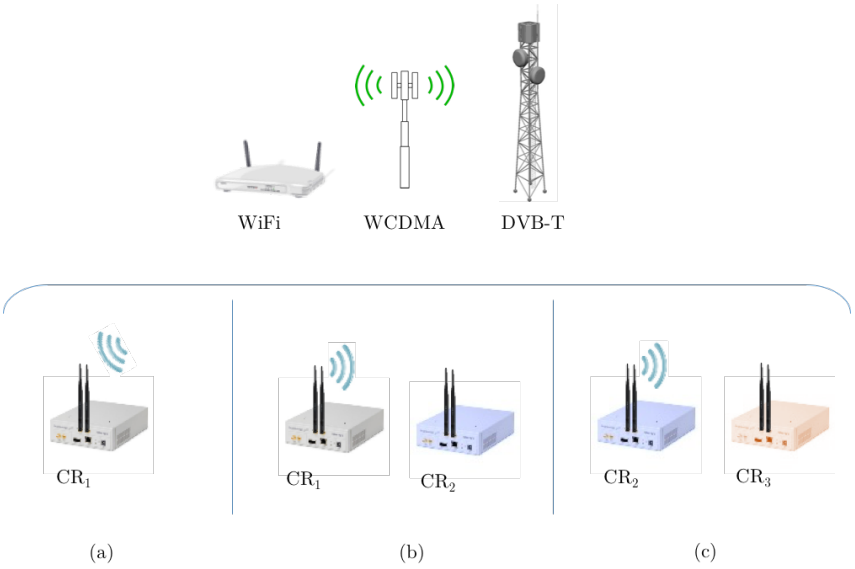


Fig. 5.26 3 Stages of learning: Learning, Teaching, Re-teaching

having already converged to the semi-optimal collision rate.

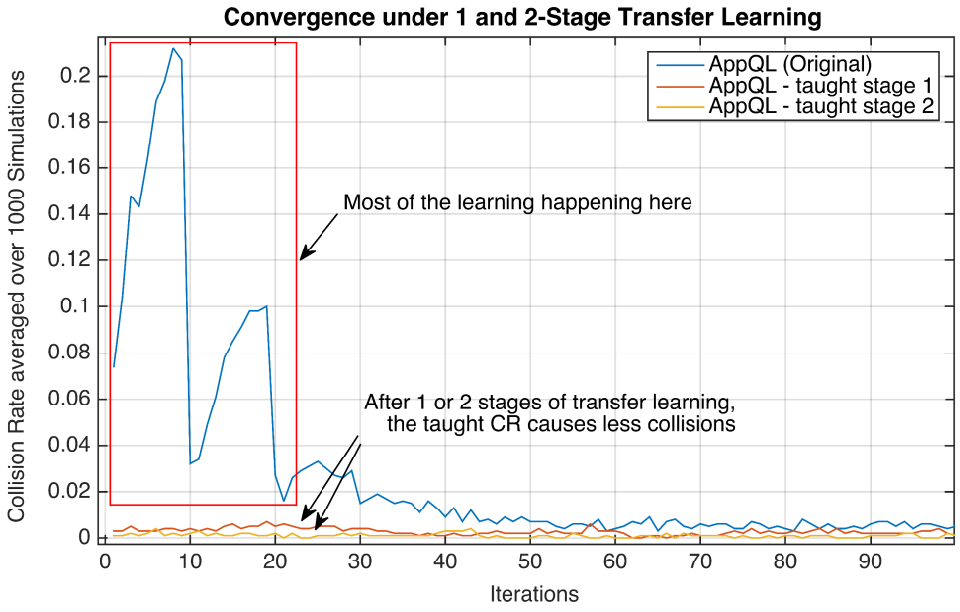


Fig. 5.27 Convergence Rate comparison between Teacher and Taught Cognitive Radios

To quantify the improvements made after one stage and two-stage transfer-learning process, Figure 5.28 shows the results in terms of primary-secondary

collisions experiences by CR_1, CR_2 and CR_3. Here, we can clearly observe two phenomena:

1. The number of collisions is drastically reduced after one learning stage,the case shown in 5.26b, such that the percentage average collisions after learning is reduced by 66%.

2. In the second stage of learning, it would not have made a significant difference whether CR_3 learned from the original teacher, CR_1, or the actual teacher, CR_2, since the improvements in collisions observed is a mere 6%.

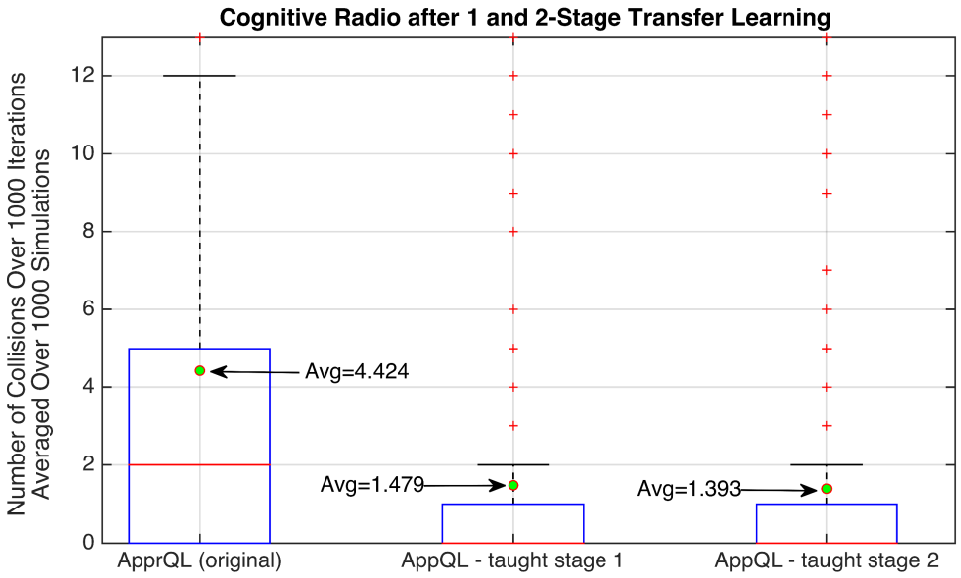


Fig. 5.28 Collision Rate before and after Transfer Learning

Another benefit of the technique can shown in terms of the overhead entailed in the information exchange. The overhead of AppQL is very low, since only the coefficients of the feature function (three coefficients, for the configuration discussed above) needs to be transferred to the naive cognitive radio, as compared to the whole Q–table as proposed by [76].

5.6 Summary and Conclusion

In this chapter, we reached several interesting conclusions, of which some did not turn out as initially envisioned. Herein, we discussed a deep learning algorithm, Sparse Stacked Auto-Encoder NN, that was used to extract features from a frequency domain of the received signals. These features were subsequently input into a reinforcement learning algorithm, which exploited this information in order to sense and access the frequency channels while trying to avoid causing interference to the primary system.

The Stacked SAE NN achieved very good results in classifying the incoming signals. In fact it outperformed our previous RAT classification attempts in Chapters 3 and 4. Likewise, the proposed Approximate Q-Learning Algorithm yielded a better performance in most of the performance analyses exercises as compared to greedy-QL, ϵ-greedy QL and the functional QL algorithms.

In RL tasks, we generally find that the choice of the optimal algorithm depends on the nature of the problem: if the channels were absolutely noiseless, then the Greedy QL algorithm would accomplish the goal of the problem only after visiting each channel once. However, this is not the case in practice, since the radio environment is chaotic and non-stationary since many different signals interact with each other simultaneously, due to the implemented Poisson packet arrival process. On the other hand, ϵ-greedy has the advantage of exploring channels in a more systematic (on average) manner and thereby would perform better in noisy radio environments as compared to a QL algorithm with greedy behavioral policy.

With regards to the functional Q-Learning form that was proposed in an attempt to make the algorithm aware of the *frequency* the channels were visited in addition to the feedback reward signal: Although this came at an additional

time and space complexity cost, we did not see a overall improvement, apart from the channel utilization diversity test. As for the proposed Approximate Q-Learning algorithm, which makes up one of the main contributions of this work, we noticed that it possesses very good features, as was tested in section 5.5. In short, it performed best in the scalability, agility, mobility, diversity (on a par with the functional QL form) and space complexity.

Here we also found out that mobility did not have a significant effect on the performance of the algorithms in terms of primary-secondary collision rate. This was found to be true since the TVWS band considered has lower frequencies, e.g. 700 MHz, than the frequency bands used for 3G systems, 1800 MHz. In other words, the Doppler spread was very low and thus did not contribute to a major change in the channel strength during the time scale of one simulation run (1000 iterations).

Chapter 6

Conclusions and Perspective

Although there has been an explosion of new techniques in AI in the past decade, the pace of CR research involving the latest techniques has been slow. Therefore, the goal of the works herein has been twofold: first, to exploit the most effective learning techniques from the field of computer science and apply them appropriately to cognitive radios; second, to explore new approaches and propose novel algorithms that combine the best of all worlds for the purpose of enhancing the performance of a CR at Radio Access Technology recognition and channel assignment tasks.

In this chapter we will draw overall conclusions from previous chapters and set the perspective from the point of view of this thesis into the future.

6.1 Conclusions

Going back to the main goals of CR technology [7–9, 12] since its inception the aspirations behind the cognitive radio technology has been *intelligent*, *autonomous* operation and *self-organization*. Therefore, the bases of this thesis,

as well, inline with that of Cognitive Radio technology, was to put in place intelligent mechanisms that are *Autonomous* and *Context-Aware*, breaking away from the traditional hard-wiring of code to do a specific task.

In this research work we started by laying out the challenges that currently obstruct the adoption of cognitive radios in a wider sense. Some of these challenges were: interference to primary users caused by coexisting secondary transmissions and lack of regulatory policy enforcement mechanism.

To address some of the above-mentioned challenges, in this thesis, we have taken the following approach. Based on the fact that we can infer the priority of a radio node's transmission license from its RAT type, we have first attempted identify different classes of Radio Access Technologies. This was the subject of Chapters 3, 4 and Section 5.3 of Chapter 5. In these works, we proposed novel RAT recognition techniques, namely, *MLP Neural Nets*, *SOFM-SVM* and *Deep SAE* classifiers, and showed their viability in severely noisy scenarios, e.g. at SNR of -25 dB. To the best knowledge of the authors, these results are currently thought to be unrivaled.

One of the key contributions of Chapters 3 and 4 is that unlike previous works in the literature where sets of selected features were manually hand engineered, in both of these chapters we designed learning algorithms that learned directly from the raw data. However, perhaps it is worth mentioning that learning from time-domain raw data was less straightforward. As we found out in Chapter 3 that fixing the time-domain sample window for the acquisition of the samples of one particular RAT introduces a wrap-around effect that leads to the shifting of the salient features from one frame to another. This ultimately made the MLP neural net training impossible at the low SNR regime. We overcame this challenge at the cost of introducing frame synchronization by using cross-correlation of subsequent frames. This lead to around 200% better

classification error as compared to matched filter baseline method at -15 dB.

One might reasonably argue that synchronization is a preprocessing stage that would come at a cost to the computational complexity of such a learning architecture. However almost all wireless receivers already perform some sort of synchronization anyway and therefore we believe its fine to assume that the data fed into the MLP architecture to be in its synchronized form.

Apart from achieving an excellent classification error rate at -25 dB SNR, the key novelty of the SOFM-SVM method proposed in Chapter 4 is that, unlike the fully supervised training method used in Chapter 3, the data is first passed through an *unsupervised* SOFM neural net that acts as non-linear generalization of PCA method [117]. The benefit of using an intermediary SOFM layer is two-folds: it is an unsupervised feature extractor; and it helps in dimensionality reduction from the number of dimensions of the raw spectrum vector to the number of neurons at the SOFM computational layer, and therefore also lowers the complexity involved in training the succeeding SVM classifier.

Chapter 5, which is the core Chapter of the thesis, showed that:

- Using contextual information extracted from the received signal, we can enhance the performance of the CR in channel assignment task, by lowering primary-secondary collision rate.
- Reinforcement Learning fits the framework of a cognitive radio operation very well due to its light-weight distributed nature. There are many types of Reinforcement Learning algorithms that could be used by a CR, however the ones that are adequate in a realistic scenario need to be model-free and off-policy.
- Through extensive simulation results using Approximate Q-learning we have proposed a new method to sense and access a candidate white-space

spectrum channel in many ways better than tabular greedy, ϵ-greedy and functional Q-learning.

- We have also been able to achieve channel assignment-related parameter compression, as was shown in the space complexity analysis in Chapter 5.
- The learning capability of AppQL was tested in Section 5.5.7, Based on the results that knowledge learned by a CR was shown to be beneficial in teaching other naive CRs, we can conclude that a learning process had taken place. However, the assumption therein shall be noted such that the taught CRs were thought to be able to understand what the feature coefficients meant, i.e. they were also making use of a linear function approximator with the same feature functions.
- We also discussed that another advantage of the AppQL method is that the reward feature can be altered on demand to reflect the rules and policies of a spectrum regulator in a particular spectrum band. Due to the short learning time of CRs equipped with such an access mechanism, new policies can be put in place in a dynamic fashion.

Our final contribution to the research community will be our real-life testbed datasets. We are planning to upload our DVB-T, WDCMA and WiFi RAT datasets of over a gigabyte in capacity online for academic research use. One of the reasons it is hard to benchmark a classification algorithm on different RATs at different SNRs is basically because (to the best of our knowledge) there is no common dataset available online to all researchers. We spent a significant amount of time and effort to come up with a discussed testbed setup presented in Chapter 2 and we think we would make it easier for fellow researchers of this topic to either benchmark their results against each other or just use it for educational purposes.

6.2 Future Works and Perspective

There are several important items discussed in this thesis that can be pursued further:

- It is possible that a complicated algorithm would do the job of the RL learning methods described in this work. However, the point in using RL is to generalize rather than design a hard-wired model to execute a certain task. In this regard, it Q–Learning with some sort of function approximation technique, easily allows for extension in terms of the feature function employed to model the state of an environment. A more profound question here, is, however, which feature functions? and how to automatically generate these functions? A possible attempt could be connecting the internal representations of a deep neural network,that are excited when they see a specific feature, to Equation 5.13 and run the Q-learning algorithm first in reverse to identify those feature functions whose coefficients, w, get higher values. This is a novel and exciting research direction that we consider to undertake in the future.

- Uniqueness: Unless two devices are precisely identical in hardware and software, it cannot be easily assumed that exchange learned knowledge will be straightforward, especially in the case of a cognitive radio, where radio agent experience matters most. Even when two CRs operate the same software, there might be subtle different radio characteristics at hardware [24] where the information taught would be interpreted differently. After all, cognitive radios are not all meant to be talking the same language, e.g. use the same RAT, yet they should be able to share learned knowledge. For instance, a WiFi and Bluetooth device coexisting in the 2.4 GHz band, should be able to share knowledge e.g. how to

avoid interference, how to improve throughput, etc. This would be an interesting issue to tackle in future research.

- Simulation time is not necessarily the same as real-time. Works published in scientific journals in wireless communications that rely on simulations, often times the model that runs on the computer does not reflect the actual physical time. This may lead to failure when an algorithm is taken from development stage to production. In this work, according to the mobility model discussed, we have stated that the 1000 iterations, that was run in every experiment, maps to around 1 msec of actual physical time. Thus we assume it is within an acceptable time-frame, since an LTE system the Transmission Time Interval (TTI) is also 1 msec. However, in order to make sure this is the case, as a future work it would be appropriate to compare the system-level simulation timing analyses presented in Chapter 5 with a on-chip timing performance of the AppQL.

Critics may argue that other recent technologies such as software defined communication may also be well fit to execute the task at hand. For example, a CR can be made to download specific instructions that would change its internal configurations such that it becomes up to date with the regulations of white-space channel access of a certain area. Although, getting frequent updates from a regulatory system and the throughput overhead that it entails in updating radio units could be an obstacle. However, against all odds, this is certainly another possible option. Taking this approach further we need deal with a massive connectivity challenge, frequent periodic data base updates, handle large volume of channel access requests and harmonize the various sources on input. This is were Big Data techniques come handy.

Big Data analytics is wrought with complexity. It must define the available data, cleanse the data, join the data with other databases, store and visualize

the data. To move from manually guiding big data analytics through each step of the process to automating each step is a considerable challenge. Previously data was considered plain IP packets as it traversed telco networks. With the advent of Big Data enabled networking, the cognitive advantage of such systems is envisioned to aid in optimizing the data flow through contextual information extracted from metadata of different industries. In Section 6.2.1, as the bases for our future works we entertain the uses of Big Data tools in Telecoms and propose a novel Big Data-based Mobile Network architecture that is designed to be able to scale with the massive inflow of data.

6.2.1 The Role of Big Data in Future Communication Systems

A part of our ongoing works that did not fully make it into this thesis is the research of applications of Big Data systems in next generation wireless communication networks. Below we briefly present an extract of these works.

Mobile RF data, similar to Big Data, is large in volume, unstructured and diverse in type, and travels at various speeds. As shown in Figure 2.5a, according to Cisco's projections [1], between 2013 and 2018, mobile network connection speeds will increase two–fold, and global mobile data traffic will increase nearly 11–fold, while it continues to grow 3 times faster than global fixed IP traffic. In addition, the types of data that will make up these figures are going to be unstructured, that is, it will constitute of many various types of raw data from documents and log data to audio and visual data [2]. Figure 2.5b illustrates some of these statistics.

However, since the time we have crossed over from a voice dominant era and entered the realm of data–oriented networking, mobile service providers are

experiencing a Cost–Revenue decoupling. As shown in Figure 6.1, costs of running a cellular system, driven by the exponential increase in data demand, have increased tremendously; however, revenues have started to plateau because of saturated markets, flat–rate tariffs, competition and regulatory pressure to make Internet access more affordable. From the network economics point of view, this commoditization of network services is a clear indicator that requires mobile operators to rethink their business models so that they can stay financially sustainable.

One of the most valuable but underused assets of a mobile communication service provider is its users' data. As millions of direct customers of a mobile operator use the network on daily bases they leave behind digital footprint and metadata that could be analyzed and turned into new revenue streams in terms of new apps and services. This is where Big RF comes in. A Big RF framework is one that is able to take advantage of the current huge traffic surge that passes through the telco's pipes, turning its passive pipes into smart pipes by applying sophisticated analytics such as Artificial Intelligence and Machine Learning techniques to the data passing through it. By applying complex analytics, a Big Data enabled system will have the potential to predict future events and thereby improve wireless networks response, e.g., aiding dynamic spectrum access for increased capacity and to achieve superior network management dynamically. The Big RF framework is foreseen to be the enabler for Self–Organizing Networks (SONs) by predicting and managing spectrum usage and also improve network performance after initial deployment [191].

Functional Architecture

Big Data in Telecoms may go back to early generations of mobile communication systems. However, at the time, neither adequate data storage nor capable

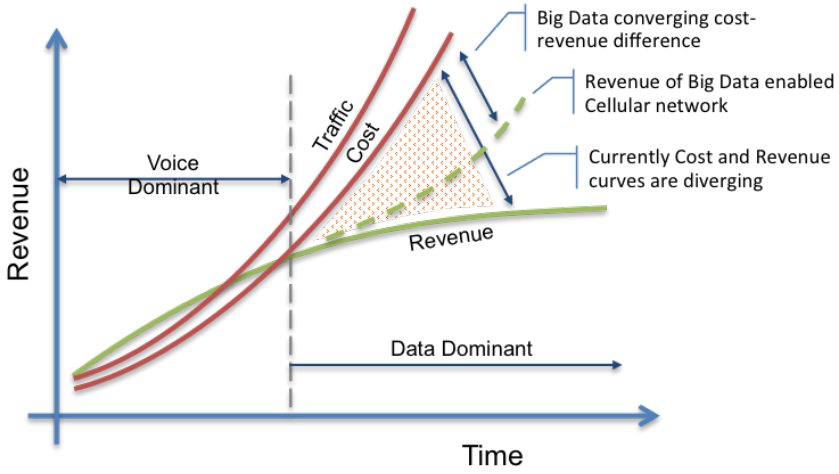


Fig. 6.1 Mobile Operator Cost–Revenue Decoupling

analytical techniques was available at the scale required. However, the development of both software and hardware technologies especially in the past decade gradually alleviated these bottlenecks paving the way towards massive distributed parallel processing paradigm in the realm of Big Data.

In general, there are two main classifications of data processing and these are batch, for relatively static data, and streamed, for relatively dynamic data, processing. Like big business, Big RF also has unique requirements that must be satisfied; Because of the velocity, variety and enormity of the amount of data ingested by Telcos, meeting this challenge requires the use of both distributed computing methods mentioned above. Figure 6.2 illustrates a novel hybrid architecture that utilizes both stream and batch processing methods to satisfy these needs. Batch processing is used to analyze imported static databases from a variety of sources, such as Regulatory and Television White Space (TVWS) database managers, while stream processing is tapped to process near real time incremental data, such as localized spectrum usage, wireless microphone registrations with a TVWS database, user behavior analysis for congestion control, etc. In the case of Big RF analytics the stream processing

is essential since the huge amount of data ingested by the network at a high velocity would result in long processing delays if conventional batch processing is used.

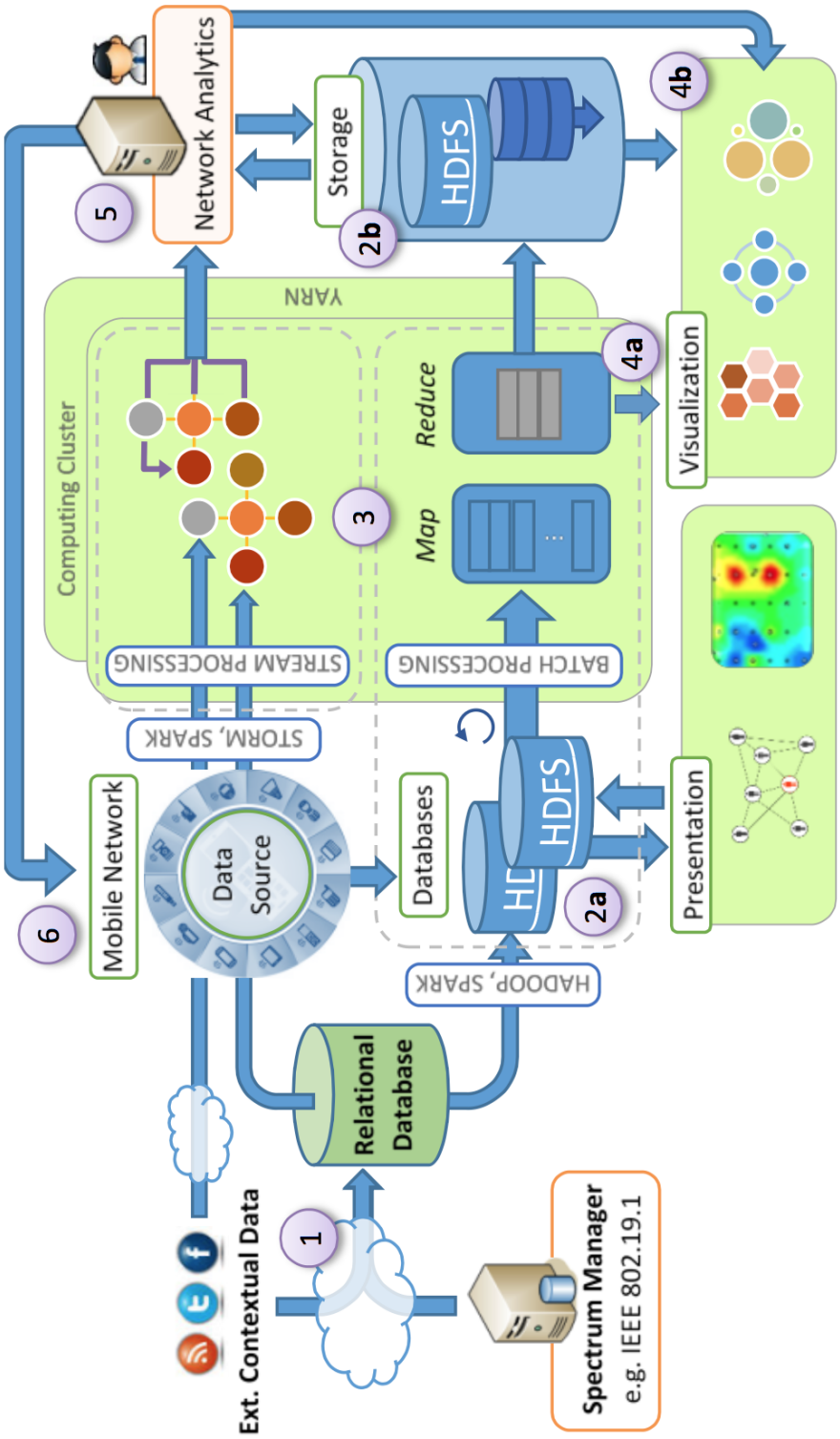


Fig. 6.2 A Big RF system functional architecture

The Big RF functional architecture illustrated in Figure 6.2 generally follows a sequence of steps from the ingestion of the raw data from various sources to producing actionable insights that reconfigures the wireless network and maintains its robustness. A list of steps corresponding to the data flow illustrated in Figure 6.2 are as follows:

1. **Ingest**: First, data sources, such as spectrum data, social media and network logs are identified. Then, depending on the type of the data, if steaming, stream links are setup, otherwise, data is queued up in a database until data batches are formed. The targeted data then can be either ingested via the Internet or an operator's private backhaul.

2. **Store**: Storage of vast amounts of ingested data happens both before and after the processing phase (step 3):

   (a) Prior to analyzing a massive dataset it is essential to curate the data, e.g. normalization and distillation, to avoid bias and reduce variance.

   (b) After the main data processing stage, the results of the computation are stored on the database again.

3. **Analyze**: The curated data is passed through multiple computing nodes of a cluster for the information to be processed according to some prescribed topology. Two noticeable processing candidates for this system would be Hadoop [192] for batch[1] data and Storm [193, 194] for streaming data. Meanwhile, Spark [195, 196] is also another cluster computing system that speeds data analytics by utilization of in–memory (RAM based) computation rather than disk based processing as with Hadoop. Spark can be used for both batch and stream distributed data processing. A

[1]Recent developments in Hadoop allow for stream processing as well.

comparison of these three well–known cluster–computing tools is given in Table 6.1. The combination of these two distributed architectures could

Table 6.1 Comparison of different distributed cluster computing systems for Big RF

	Language	RAM Based	Processing mode	Latency	Application
Hadoop	Java	No	Batch	Slow	General purpose
Storm	Clojure and Java	No	Stream	Depends	Good at processing small incremental incoming data at Near Real–Time
Spark	Scala, supports Java and Python	Yes	Batch and Stream	Fast	Best for iterative algorithms, and interactive data mining

be collocated on the same computer cluster through YARN (Yet Another Resource Negotiator), which is a cluster management tool. Hadoop is the most well–known distributed batch processing platform and is supported by the Hadoop Distributed File System (HDFS) for managing massive number of data files, and the MapReduce runtime system that provides the scalable foundations of distributed analytics over the data in addition to numerous other tools. Storm, on the other hand, is an event–based stream processing system where data streams flow through multi–nodal topologies, and each node performs sophisticated data processing with the results saved on a storage system.

4. **Visualization**: In order to gain technical or business insight from the processed data and make better decisions, visualization is an essential tool. Data visualization can happen at two stages:

   (a) During data analysis: to gain immediate feedback, especially if the data processing time is on the order of tens of minutes or higher.

   (b) While data is parked either before or after processing step (Step 3).

5. **Network Intelligence**: Predictive and retrospective analyses are key to delivery of Big RF. After the unstructured raw data is processed, the knowledge that it generated need to be turned into specific machine–understandable actions. At this step, final analysis can be applied to learn from the subscribers past behavior where users network performance metrics could be tracked to identify faults in the mobile Radio Access Network (RAN) and fix them.

6. **Predictive Models**: Applying Predictive Models, which are advanced statistical or machine learning methods that are trained on past data to project future data, is also desired so that the system can predict and avoid possible future network failures, such as network congestion due to an accident or massive event.

Traditionally, carriers' network operations were based on custom made hardware mostly running proprietary software, which often led to a vendor locked situation. Furthermore, since an operator's equipment, e.g., base stations, were distributed over a geographical area, each computing unit's power supply, cooling system, site rental and maintenance had to be managed separately [197]. Having foreseen these difficulties, carriers are increasingly moving operations into the cloud to better allocate and manage resources, and reduce geographic–footprint costs. Although, a decade ago the distributed architectures were ascendant, e.g., giving more responsibility to distributed eNodeBs, today, centralization is becoming increasingly popular. The rethinking of carriers' core network and last mile architectures has led to two key paradigms: the Carrier Cloud (C–Cloud), and the Cloud–Radio Access Network (C–RAN).

Looking into the future, we find that the state-of-the-art data center and networking technologies, such as SDN, NFV (Network Function Virtualization) and SON, to be inherently based on the dynamics of the network. Therefore,

we believe that data-driven Big RF–like solutions will be one of the main pillars to drive the profitability of telecoms industry by adding intelligence on how the network handles Big Data. Realizing such an automated and intelligent self–organizing, self-healing solution is an important area of research that we will further explore. In particular, we are interested in further researching the following three possible advantages:

- Offloading computation to the local BBU (Baseband Unit): Information processing on the cached data offloads communication and processing overhead that would otherwise be required if one main cloud computing system was used.
- Enhanced Security and Privacy: Localizing information reduces the attractiveness of any one target and reduces the number of potentially interceptable information transfers.
- Reduced Latency: For those applications that require only local communications, running algorithms locally on the data cached at the BBU reduces the end–to–end RTT delay [198].

References

[1] Cisco . Cisco visual networking index: Forecast and methodology, 2014–2019. *CISCO White paper*, 2015.

[2] Bob Fox, Rob van den Dam, and Rebecca Shockley. Analytics: Real-World Use of Big Data in Telecommunications. *IBM Institute for Business Value*, 2013.

[3] ET FCC. Docket No 03-222 Notice of proposed rule making and order. Technical report, December, 2003.

[4] UK Ofcom. Implementing geolocation. Technical report, 2015-Feb.

[5] UK Ofcom. Regulatory Requirements for White Space Device in the UHF TV Band. 2012.

[6] Stephan Monterde. Cisco Technology Radar. Technical report, Cisco, 2015.

[7] Ian F Akyildiz, Won-Yeol Lee, Mehmet C Vuran, and Shantidev Mohanty. NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey. *Computer Networks*, 50(13):2127–2159, 2006.

[8] Joseph Mitola III. Cognitive radio for flexible mobile multimedia communications. In *Mobile Multimedia Communications, 1999.(MoMuC'99) 1999 IEEE International Workshop on*, pages 3–10. IEEE, 1999.

[9] Joseph Mitola III and Gerald Q Maguire Jr. Cognitive radio: making software radios more personal. *Personal Communications, IEEE*, 6(4):13–18, 1999.

[10] Joseph Mitola III. *Cognitive Radio – An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, Royal Institute of Technology (KTH), 2000.

[11] Joe Mitola. The software radio architecture. *Communications Magazine, IEEE*, 33(5):26–38, 1995.

[12] Simon Haykin. Cognitive radio: brain-empowered wireless communications. *Selected Areas in Communications, IEEE Journal on*, 23(2):201–220, 2005.

[13] FCC. FCC Releases Rules for Innovative Spectrum Sharing in 3.5 GHz Band. 2015.

[14] Ekram Hossain, Mehdi Rasti, Hina Tabassum, and Amr Abdelnasser. Evolution toward 5G multi-tier cellular wireless networks: An interference management perspective. *Wireless Communications, IEEE*, 21(3):118–127, 2014.

[15] Jeffrey G. Andrews, Stefano Buzzi, Wan Choi, Stephen V. Hanly, Aurelie Lozano, Anthony CK Soong, and Jianzhong Charlie Zhang. What will 5G be? *Selected Areas in Communications, IEEE Journal on*, 32(6):1065–1082, 2014.

[16] GSMA Intelligence. Understanding 5G: Perspectives on future technological advancements in mobile. *GSMA Intelligence Understanding 5G*, pages 3–15, 2014.

[17] Przemyslaw Pawelczak, Keith Nolan, Linda Doyle, Ser Wah Oh, and Danijela Cabric. Cognitive radio: Ten years of experimentation and development. *Communications Magazine, IEEE*, 49(3):90–100, 2011.

[18] Erich Stuntebeck, Timothy O'Shea, Joseph Hecker, and T. Clancy. Architecture for an open-source cognitive radio. In *Proceedings of the SDR forum technical conference*, 2006.

[19] Ian F. Akyildiz, Won-Yeol Lee, and Kaushik R. Chowdhury. CRAHNs: Cognitive radio ad hoc networks. *AD hoc networks*, 7(5):810–836, 2009.

[20] Paul Sutton, Jorg Lotze, Hicham Lahlou, Suhaib Fahmy, Keith Nolan, Baris Ozgul, Thomas W. Rondeau, Juanjo Noguera, Linda E. Doyle, and others. Iris: an architecture for cognitive radio networking testbeds. *Communications Magazine, IEEE*, 48(9):114–122, 2010.

[21] S. Baban, D. Denkoviski, O. Holland, L. Gavrilovska, and H. Aghvami. Radio access technology classification for cognitive radio networks. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on*, pages 2718–2722, September 2013.

[22] Ioannis Dagres, Andreas Polydoros, Daniel Denkovski, Valentin Rakovic, Vladimir Atanasovski, Liljana Gavrilovska, Krzysztof Cichon, Adrian Kliks, Shaswar Baban, and Oliver Holland. An Integrated Platform for Source Detection, Identification and Localization with Applications to Cognitive-Radio. In *Wireless Conference (EW), Proceedings of the 2013 19th European*, pages 1–6, April 2013.

[23] Shaswar Baban, Daniel Denkoviski, Oliver Holland, Liljana Gavrilovska, and Hamid Aghvami. Radio Access Technology Classification for Overlay and Underlay Cognitive Radio Networks. In *SDR WInnComm'12 - 2nd Annual Acropolis Workshop*, 2012.

[24] Shaswar Baban, Oliver Holland, and Hamid Aghvami. Wireless Standard Classification in Cognitive Radio Networks Using Self-Organizing Maps. In *Wireless Communication Systems (ISWCS 2013), Proceedings of the Tenth International Symposium on*, pages 1–5, August 2013.

[25] Shaswar Baban and Hamid Aghvami. RAT-Aware Cognitive Radio Spectrum Access: Using Deep Learning For Reduced Primary-Secondary Collision Rate. Technical report, 2015. Published: [Submitted].

[26] James Neel, Peter G. Cook, Ihsan Akbar, Neal Mellen, Shaswar Baban, Charles Sheehe, Bob Schutz, and Daniel Devasirvatham. IPA Volume 3: Cognitive Radio Context, WISDM, and Big RF. In *SDR-WInnComm*, 2014.

[27] J. Neel, Shaswar Baban, P. Cook, N. Mellen, I. Akbar, D. Devasirvatham, C. Sheehe, and B. Schutz. Context-Aware Cognitive Radio for Automated Wireless System Management,. In *SDR-WInnComm*, 2014.

[28] J. Neel, S. Baban, N. Mellen, I. Akbar, C. Sheehe, B. Schutz, and P. Cook. Big RF for homeland security applications. In *Technologies for Homeland Security (HST), 2015 IEEE International Symposium on*, pages 1–6, April 2015.

[29] J. Neel, Shaswar Baban, P. Cook, I. Akbar, N. Mellen, C. Sheehe, and D. Devasirvatham. Big RF for Management of Shared Spectrum Networks. In *Wireless Innovations' Forum on Wireless Communication Technologies and Software Defined Radio (WInnComm)*, 2015.

[30] techUK. 5G Innovation Opportunities. Technical report, techUK, October 2015.

[31] Harry Urkowitz. Energy detection of unknown deterministic signals. *Proceedings of the IEEE*, 55(4):523–531, 1967.

[32] Beibei Wang and K. J. Liu. Advances in cognitive radio networks: A survey. *Selected Topics in Signal Processing, IEEE Journal of*, 5(1):5–23, 2011.

[33] Jianfeng Wang, Monisha Ghosh, and Kiran Challapali. Emerging cognitive radio applications: A survey. *Communications Magazine, IEEE*, 49(3):74–81, 2011.

[34] Ian F. Akyildiz, Won-Yeol Lee, Mehmet C. Vuran, and Shantidev Mohanty. A survey on spectrum management in cognitive radio networks. *Communications Magazine, IEEE*, 46(4):40–48, 2008.

[35] Mario Bkassiny, Yang Li, and Sudharman K Jayaweera. A survey on machine-learning techniques in cognitive radios. *Communications Surveys & Tutorials, IEEE*, 15(3):1136–1159, 2013.

[36] Charles Clancy, Joe Hecker, Erich Stuntebeck, and Tim O. Shea. Applications of machine learning to cognitive radio networks. *Wireless Communications, IEEE*, 14(4):47–52, 2007.

[37] An He, Kyung Kyoon Bae, Timothy R Newman, Joseph Gaeddert, Kyouwoong Kim, Rekha Menon, Lizdabel Morales-Tirado, James Jody Neel, Youping Zhao, Jeffrey H Reed, and others. A survey of artificial intelligence for cognitive radios. *Vehicular Technology, IEEE Transactions on*, 59(4):1578–1592, 2010.

[38] Maziar Nekovee. Quantifying the availability of TV white spaces for cognitive radio operation in the UK. In *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, pages 1–5. IEEE, 2009.

[39] Maziar Nekovee. A survey of cognitive radio access to TV white spaces. In *Ultra Modern Telecommunications & Workshops, 2009. ICUMT'09. International Conference on*, pages 1–8. IEEE, 2009.

[40] Maziar Nekovee, Tim Irnich, and Jörgen Karlsson. Worldwide trends in regulation of secondary access to white spaces using cognitive radio. *Wireless Communications, IEEE*, 19(4):32–40, 2012.

[41] Tevfik Yücek and Hüseyin Arslan. A survey of spectrum sensing algorithms for cognitive radio applications. *Communications Surveys & Tutorials, IEEE*, 11(1):116–130, 2009.

[42] Hongjian Sun, Arumugam Nallanathan, Cheng-Xiang Wang, and Yunfei Chen. Wideband spectrum sensing for cognitive radio networks: a survey. *Wireless Communications, IEEE*, 20(2):74–81, 2013.

[43] Shree Krishna Sharma, Tadilo Endeshaw Bogale, Symeon Chatzinotas, Bjorn Ottersten, Long Bao Le, and Xianbin Wang. Cognitive radio techniques under practical imperfections: A survey. *Communications Surveys & Tutorials, IEEE*, 17(4):1858–1884, 2015.

[44] Claudia Cormio and Kaushik R. Chowdhury. A survey on MAC protocols for cognitive radio networks. *Ad Hoc Networks*, 7(7):1315–1329, 2009.

[45] Antonio De Domenico, Emilio Calvanese Strinati, and Maria-Gabriella Di Benedetto. A survey on MAC strategies for cognitive radio networks. *Communications Surveys & Tutorials, IEEE*, 14(1):21–44, 2012.

[46] T. Vamsi Krishna and Amitabha Das. A survey on MAC protocols in OSA networks. *Computer Networks*, 53(9):1377–1394, 2009.

[47] Alexandros G. Fragkiadakis, Elias Z. Tragos, and Ioannis G. Askoxylakis. A survey on security threats and detection techniques in cognitive radio networks. *Communications Surveys & Tutorials, IEEE*, 15(1):428–445, 2013.

[48] Gianmarco Baldini, Taj Sturman, Abdur Rahim Biswas, Ruediger Leschhorn, Gyözö Gódor, and Michael Street. Security aspects in software defined radio and cognitive radio networks: a survey and a way ahead. *Communications Surveys & Tutorials, IEEE*, 14(2):355–379, 2012.

[49] Alireza Attar, Helen Tang, Athanasios V. Vasilakos, F. Richard Yu, and Victor Leung. A survey of security challenges in cognitive radio networks: Solutions and future research directions. *Proceedings of the IEEE*, 100(12):3172–3186, 2012.

[50] Rajesh K. Sharma and Danda B. Rawat. Advances on security threats and countermeasures for cognitive radio networks: A survey. *Communications Surveys & Tutorials, IEEE*, 17(2):1023–1043, 2015.

[51] Maria-Gabriella Di Benedetto, Stefano Boldrini, Carmen Juana Martin Martin, and Jesus Roldan Diaz. Automatic network recognition by feature extraction: a case study in the ISM band. In *Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM), 2010 Proceedings of the Fifth International Conference on*, pages 1–5. IEEE, 2010.

[52] Mohammad Bari, Awais Khawar, T. Charles Clancy, and others. Recognizing FM, BPSK and 16-QAM using supervised and unsupervised learning techniques. In *2015 49th Asilomar Conference on Signals, Systems and Computers*, pages 160–163. IEEE, 2015.

[53] J. J. Popoola and R. van Olst. Application of neural network for sensing primary radio signals in a cognitive radio environment. In *AFRICON, 2011*, pages 1–6, September 2011.

[54] Jide Popoola and Rex Olst. A novel modulation-sensing method. *IEEE Vehicular Technology Magazine*, 3(6):60–69, 2011.

[55] Timothy J. O'Shea and Johnathan Corgan. Convolutional radio modulation recognition networks. *arXiv preprint arXiv:1602.04105*, 2016.

[56] Timothy J. O'Shea, Latha Pemula, Dhruv Batra, and T. Charles Clancy. Radio Transformer Networks: Attention Models for Learning to Synchronize in Wireless Systems. *arXiv preprint arXiv:1605.00716*, 2016.

[57] Timothy J. O'Shea, Johnathan Corgan, and T. Charles Clancy. Unsupervised Representation Learning of Structured Radio Communication Signals. *arXiv preprint arXiv:1604.07078*, 2016.

[58] William A. Gardner. Exploitation of spectral redundancy in cyclostationary signals. *Signal Processing Magazine, IEEE*, 8(2):14–36, 1991.

[59] Scott Enserink and Douglas Cochran. On detection of cyclostationary signals. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 3, pages 2004–2007. IEEE, 1995.

[60] Kiran Joshi, Steven Hong, and Sachin Katti. Pinpoint: Localizing interfering radios. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 241–253, 2013.

[61] Osamu Ohno, Masaaki Katayama, Takaya Yamazato, and Akira Ogawa. A simple model of cyclostationary power-line noise for communication systems. In *Proceedings of the International Symposium on Power Line Communications and its Applications (ISPLC)*, pages 115–122, 1998.

[62] Jarmo Lundén, Visa Koivunen, Anu Huttunen, and H. Vincent Poor. Spectrum sensing in cognitive radios based on multiple cyclic frequencies. In *Cognitive Radio Oriented Wireless Networks and Communications, 2007. CrownCom 2007. 2nd International Conference on*, pages 37–43. IEEE, 2007.

[63] Paul Sutton, Keith Nolan, and Linda E. Doyle. Cyclostationary signatures for rendezvous in ofdm-based dynamic spectrum access networks. In *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, pages 220–231. IEEE, 2007.

[64] Paul D. Sutton, Keith E. Nolan, and Linda E. Doyle. Cyclostationary signatures in practical cognitive radio applications. *Selected Areas in Communications, IEEE Journal on*, 26(1):13–24, 2008.

[65] Simon Haykin, David J. Thomson, and Jeffrey H. Reed. Spectrum sensing for cognitive radio. *Proceedings of the IEEE*, 97(5):849–877, 2009.

[66] Jarmo Lundén, Saleem A. Kassam, and Visa Koivunen. Robust nonparametric cyclic correlation-based spectrum sensing for cognitive radio. *Signal Processing, IEEE Transactions on*, 58(1):38–52, 2010.

[67] Ala' Al-Habashna, Octavia Dobre, Ramachandran Venkatesan, Dimitrie C Popescu, and others. Second-order cyclostationarity of mobile WiMAX and LTE OFDM signals and application to spectrum awareness in cognitive radio systems. *Selected Topics in Signal Processing, IEEE Journal of*, 6(1):26–42, 2012.

[68] Paul D. Sutton, Baris Özgül, and L. E. Doyle. Cyclostationary signatures for LTE Advanced and beyond. *Physical Communication*, 10:179–189, 2014.

[69] K. Kim, I. A. Akbar, K. K. Bae, J. S. Um, C. M. Spooner, and J. H. Reed. Cyclostationary Approaches to Signal Detection and Classification in Cognitive Radio. In *2007 2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, pages 212–215, April 2007.

[70] Berna Sayrac, Halina Uryga, Wladimir Bocquet, Pascal Cordier, and Sebastien Grimoud. Cognitive radio systems specific for IMT systems: Operator's view and perspectives. *Telecommunications Policy*, 37(2):154–166, 2013.

[71] René Wörfel and Hans Dodel†. Agenda Items der WRC-12. In *Satellitenfrequenzkoordinierung*, pages 401–403. Springer Berlin Heidelberg, 2012.

[72] Hitoshi Yoshino. ITU-R Standardization Activities on Cognitive Radio. *IEICE TRANSACTIONS on Communications*, E95-B(4):1036–1043, April 2012.

[73] CORDIS Archive : European Commission : CORDIS : FP7 : ICT. http://cordis.europa.eu/fp7/ict/home_en.html, 2015.

[74] David Soldani and Antonio Manzalini. Horizon 2020 and beyond: on the 5G operating system for a true digital society. *Vehicular Technology Magazine, IEEE*, 10(1):32–42, 2015.

[75] Dennis J Baker and Anthony Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *Communications, IEEE Transactions on*, 29(11):1694–1701, 1981.

[76] Lorenza Giupponi, Ana Galindo-Serrano, Pol Blasco, and Mischa Dohler. Docitive networks: an emerging paradigm for dynamic spectrum management [dynamic spectrum management]. *Wireless Communications, IEEE*, 17(4):47–54, 2010.

[77] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, December 1943.

[78] Donald Olding Hebb. *The organization of behavior: A neuropsychological approach*. John Wiley & Sons, 1949.

[79] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[80] Marvin Minsky and Seymour Papert. *Perceptrons*. MIT press, 1969.

[81] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

[82] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, January 2015.

[83] S. Linnainmaa. The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. *Master's Thesis, Department of Computer Science, University of Helsinki*, 1970.

[84] Paul J. Werbos. Applications of advances in nonlinear sensitivity analysis. In *System modeling and optimization*, pages 762–770. Springer, 1982.

[85] Yu-Jie Tang, Qin-Yu Zhang, and Wei Lin. Artificial neural network based spectrum sensing method for cognitive radio. In *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, pages 1–4. IEEE, 2010.

[86] Bin Le, Thomas W. Rondeau, and Charles W. Bostian. Cognitive radio realities. *Wireless Communications and Mobile Computing*, 7(9):1037–1048, 2007.

[87] Liang Yin, SiXing Yin, Weijun Hong, and ShuFang Li. Spectrum behavior learning in cognitive radio based on artificial neural network. In *Military Communications Conference, 2011-Milcom 2011*, pages 25–30. IEEE, 2011.

[88] Nicola Baldo, Bheemarjuna Reddy Tamma, B. S. Manoj, Ramesh Rao, and Michele Zorzi. A neural network based cognitive controller for dynamic channel selection. In *Communications, 2009. ICC'09. IEEE International Conference on*, pages 1–5. IEEE, 2009.

[89] Jacques Palicot and Christian Roland. A new concept for wireless reconfigurable receivers. *Communications Magazine, IEEE*, 41(7):124–132, 2003.

[90] Thomas P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon. Accelerating the convergence of the back-propagation method. *Biological cybernetics*, 59(4-5):257–263, 1988.

[91] MATLAB User's Guide. The mathworks. *Inc., Natick, MA*, 5:333, 1998.

[92] Howard Demuth, Mark Beale, and Math Works. *MATLAB: Neural Network Toolbox: User's Guide*. Math Works, 2014.

[93] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[94] Douglas M Kline and Victor L Berardi. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing & Applications*, 14(4):310–318, 2005.

[95] George E Nasr, EA Badr, and C Joun. Cross Entropy Error Function in Neural Networks: Forecasting Gasoline Demand. In *FLAIRS Conference*, pages 381–384, 2002.

[96] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[97] D. Randall Wilson and Tony R. Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10):1429–1451, 2003.

[98] Yoshua Bengio, Ian J. Goodfellow, and Aaron Courville. Deep Learning. Book in preparation for MIT Press, 2015.

[99] Geoffrey E Hinton, Terrence J Sejnowski, and David H Ackley. *Boltzmann machines: Constraint satisfaction networks that learn*. Carnegie-Mellon University, Department of Computer Science Pittsburgh, PA, 1984.

[100] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural Networks: Tricks of the Trade*, pages 599–619. Springer, 2012.

[101] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[102] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[103] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[104] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, and others. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[105] Yoshua Bengio, Yann LeCun, and others. Scaling learning algorithms towards AI. *Large-scale kernel machines*, 34(5), 2007.

[106] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits. 1998.

[107] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[108] Yoshua Bengio, Ian J. Goodfellow, and Aaron Courville. Deep Learning. Book in preparation for MIT Press, 2015.

[109] Yuanhao Cui, Xiao jun Jing, Songlin Sun, Xiaohan Wang, Dongmei Cheng, and Hai Huang. Deep learning based primary user classification in Cognitive Radios. In *2015 15th International Symposium on Communications and Information Technologies (ISCIT)*, pages 165–168. IEEE, 2015.

[110] Leonardo Badia, Daniele Munaretto, Alberto Testolin, Andrea Zanella, and Michele Zorzi. Cognition-based networks: Applying cognitive science to multimedia wireless networking. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*, pages 1–6. IEEE, 2014.

[111] Michele Zorzi, Andrea Zanella, Alberto Testolin, Michele De Filippo De Grazia, and Marco Zorzi. Cognition-Based Networks: A New Perspective on Network Optimization Using Learning and Distributed Intelligence. *IEEE Access*, 3:1512–1530, 2015.

[112] Ahmed Mohamedou, Aduwati Sali, Borhanuddin Ali, and Mohamed Othman. Dynamical Spectrum Sharing and Medium Access Control for Heterogeneous Cognitive Radio Networks. *International Journal of Distributed Sensor Networks*, 2016, 2016.

[113] Alexander M Wyglinski, Maziar Nekovee, and Thomas Hou. *Cognitive radio communications and networks: principles and practice*. Academic Press, 2009.

[114] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research*, 37(23):3311–3325, 1997.

[115] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 833–840, 2011.

[116] Teuvo Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1–6, 1998.

[117] Simon S Haykin. *Neural networks and learning machines*, volume 3. Pearson Education Upper Saddle River, 2009.

[118] Teuvo Kohonen. Essentials of the self-organizing map. *Neural Networks*, 37:52–65, 2013.

[119] Guido Deboeck and Teuvo Kohonen. *Visual explorations in finance: with self-organizing maps*. Springer Science & Business Media, 2013.

[120] Pablo Tamayo, Donna Slonim, Jill Mesirov, Qing Zhu, Sutisak Kitareewan, Ethan Dmitrovsky, Eric S. Lander, and Todd R. Golub. Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proceedings of the National Academy of Sciences*, 96(6):2907–2912, 1999.

[121] Tae-Soo Chon. Self-organizing maps applied to ecological sciences. *Ecological Informatics*, 6(1):50–61, 2011.

[122] Qiao Cai, Sheng Chen, Xiaochen Li, Nansai Hu, Haibo He, Yu-Dong Yao, and Joseph Mitola. An integrated incremental self-organizing map and hierarchical neural network approach for cognitive radio learning. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–6. IEEE, 2010.

[123] MI Taj, M Akil, and O Hammami. Standard recognising self organizing map based cognitive radio transceiver. In *Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM), 2010 Proceedings of the Fifth International Conference on*, pages 1–5. IEEE, 2010.

[124] Awais Khawar and T Clancy. Signal classifiers using self-organizing maps: Performance and robustness. In *SDR Forum Technical Conference (SDR'09)*, 2009.

[125] T Charles Clancy, Awais Khawar, and Timothy R Newman. Robust signal classification using unsupervised learning. *Wireless Communications, IEEE Transactions on*, 10(4):1289–1299, 2011.

[126] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[127] Corinna Cortes and Vladimir Vapnik. Support vector machine. *Machine learning*, 20(3):273–297, 1995.

[128] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2009.

[129] Hao Hu, Junde Song, and Yujing Wang. Signal classification based on spectral correlation analysis and SVM in cognitive radio. In *Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on*, pages 883–887. IEEE, 2008.

[130] L. Gueguen and B. Sayrac. Radio access technology recognition by classification of low temporal resolution power spectrum measurements. *Wireless Communications and Mobile Computing*, 10(8):1033–1044, 2010.

[131] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[132] J. Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[133] MH Beale, MT Hagan, and HB Demuth. MATLAB Neural Network Toolbox™ User's Guide, MathWorks, Version 8.0. 1 (Release 2013a). 2013.

[134] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

[135] Csaba Szepesvari. *Algorithms for Reinforcement Learning*. Morgan and Claypool, 2010.

[136] Andrew Y. Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer, 2006.

[137] Gerald Tesauro. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[138] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. Robocup: The robot world cup initiative. In *Proceedings of the first international conference on Autonomous agents*, pages 340–347. ACM, 1997.

[139] Peter Stone, Richard S. Sutton, and Gregory Kuhlmann. Reinforcement learning for robocup soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

[140] Alexander A. Sherstov and Peter Stone. Three automated stock-trading agents: A comparative study. In *Agent-Mediated Electronic Commerce VI. Theories for and Engineering of Distributed Mechanisms and Systems*, pages 173–187. Springer, 2005.

[141] John Moody and Matthew Saffell. Learning to trade via direct reinforcement. *Neural Networks, IEEE Transactions on*, 12(4):875–889, 2001.

[142] James J. Choi, David Laibson, Brigitte C. Madrian, and Andrew Metrick. Reinforcement learning and savings behavior. *The Journal of finance*, 64(6):2515–2534, 2009.

[143] Shinji Yamamoto, Mitsuomi Matsumoto, Yukio Tateno, Takeshi Iinuma, and Toru Matsumoto. Quoit filter-A new filter based on mathematical morphology to extract the isolated shadow, and its application to automatic detection of lung cancer in X-ray CT. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 2, pages 3–7. IEEE, 1996.

[144] Subhasis Chaudhuri, Shankar Chatterjee, Norman Katz, Mark Nelson, and Michael Goldbaum. Detection of blood vessels in retinal images using two-dimensional matched filters. *Medical Imaging, IEEE Transactions on*, 8(3):263–269, 1989.

[145] I Skolnik Merrill. *Introduction to radar systems*. Mc Grow-Hill, 2001.

[146] David K Barton. *Radar system analysis and modeling*, volume 1. Artech House, 2004.

[147] Bernard F Schutz. Gravitational wave astronomy. *Classical and Quantum Gravity*, 16(12A):A131, 1999.

[148] Bernard Sklar. *Digital communications*, volume 2. Prentice Hall NJ, 2001.

[149] John G Proakis, Masoud Salehi, Ning Zhou, and Xiaofeng Li. *Communication systems engineering*, volume 94. Prentice Hall New Jersey, 1994.

[150] Andreas F Molisch. *Wireless communications*. John Wiley & Sons, 2007.

[151] Kwang-Cheng Chen and Ramjee Prasad. *Cognitive radio networks*. John Wiley & Sons, 2009.

[152] Danijela Cabric, Shridhar Mubaraq Mishra, and Robert W Brodersen. Implementation issues in spectrum sensing for cognitive radios. In *Signals, systems and computers, 2004. Conference record of the thirty-eighth Asilomar conference on*, volume 1, pages 772–776. IEEE, 2004.

[153] Arthur D Ericsson. Little and Chalmers University of Technology. *The Socio-economic Effects of Broadband Speed Upgrades as featured in Broadband Commission*, 2013.

[154] Woon Hau Chin, Zhong Fan, and Robert Haines. Emerging technologies and research challenges for 5G wireless networks. *Wireless Communications, IEEE*, 21(2):106–112, 2014.

[155] Robert Qiu and Michael Wicks. *Cognitive Networked Sensing and Big Data*. Springer, 2014.

[156] Daisuke Takaishi, Hiroki Nishiyama, Nei Kato, and Ryu Miura. Toward energy efficient big data gathering in densely distributed sensor networks. *Emerging Topics in Computing, IEEE Transactions on*, 2(3):388–397, 2014.

[157] Ankur Omar. Improving Data Extraction Efficiency of Cache Nodes in Cognitive Radio Networks Using Big Data Analysis. In *Next Generation Mobile Applications, Services and Technologies, 2015 9th International Conference on*, pages 305–310. IEEE, 2015.

[158] ACROPOLIS: Advanced coexistence technologies for radio optimisation in licensed and unlicensed spectrum.

[159] David M Skapura. *Building neural networks*. Addison-Wesley Professional, 1996.

[160] Brian J Taylor, Marjorie A Darrah, and Christina D Moats. Verification and validation of neural networks: a sampling of research in progress. In *AeroSense 2003*, pages 8–16. International Society for Optics and Photonics, 2003.

[161] Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533, 1993.

[162] Tshilidzi Marwala. *Finite element model updating using computational intelligence techniques: applications to structural dynamics*. Springer Science & Business Media, 2010.

[163] Philip M Woodward. *Probability and Information Theory, with Applications to Radar: International Series of Monographs on Electronics and Instrumentation*, volume 3. Elsevier, 2014.

[164] M.Z. Win, P.C. Pinto, and L.A. Shepp. A Mathematical Theory of Network Interference and Its Applications. *Proceedings of the IEEE*, 97(2):205 –230, February 2009.

[165] T Charles Clancy and Nathan Goergen. Security in cognitive radio networks: Threats and mitigation. In *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, pages 1–8. IEEE, 2008.

[166] S. Filin, T. Baykas, H. Harada, F. Kojima, and H. Yano. IEEE Standard 802.19.1 for TV white space coexis tence. *IEEE Communications Magazine*, 54(3):22–26, March 2016.

[167] Konstantinos Katzis and Hamed Ahmadi. Challenges Implementing Internet of Things (IoT) Using Cognitive Radio Capabilities in 5G Mobile Networks. In *Internet of Things (IoT) in 5G Mobile Technologies*, pages 55–76. Springer, 2016.

[168] Peter D. Welch. The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on audio and electroacoustics*, 15(2):70–73, 1967.

[169] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[170] Stuart Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Prentice Hall, 2010.

[171] Tim Forde and Linda Doyle. A TV whitespace ecosystem for licensed cognitive radio. *Telecommunications Policy*, 37(2):130–139, 2013.

[172] Francisco S Melo, Sean P Meyn, and M Isabel Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*, pages 664–671. ACM, 2008.

[173] FCC. FCC Releases Rules for Innovative Spectrum Sharing in 3.5 GHz Band. Technical report, 2015.

[174] Jeffrey G. Andrews. Seven ways that HetNets are a cellular paradigm shift. *Communications Magazine, IEEE*, 51(3):136–144, 2013.

[175] Andrew Ng, Jiquan Ngiam, Chuan Y Foo, Yifan Mai, and Caroline Suen. UFLDL tutorial, 2012.

[176] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[177] John N Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *Automatic Control, IEEE Transactions on*, 42(5):674–690, 1997.

[178] Technical Specification Group Radio Access Networks. 3GPP TR 25.943 V6.0.0 (2004-12). Technical report, 3GPP, December 2004.

[179] Luis M Correia. *Wireless flexible personalized communications*. John Wiley & Sons, Inc., 2001.

[180] Hamid R Maei, Csaba Szepesvári, Shalabh Bhatnagar, and Richard S Sutton. Toward off-policy learning control with function approximation. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 719–726, 2010.

[181] Richard S Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 993–1000. ACM, 2009.

[182] Richard S Sutton, Hamid R Maei, and Csaba Szepesvári. A Convergent O (n) Temporal-difference Algorithm for Off-policy Learning with Linear Function Approximation. In *Advances in neural information processing systems*, pages 1609–1616, 2009.

[183] Dimitri P. Bertsekas. *Dynamic programming and optimal control*. Number 2. Athena Scientific, 1995.

[184] Matthew E. Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, 10:1633–1685, 2009.

[185] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[186] Qiyang Zhao and David Grace. Agent transfer learning for cognitive resource management on multi-hop backhaul networks. In *Future Network and Mobile Summit (FutureNetworkSummit), 2013*, pages 1–10. IEEE, 2013.

[187] Qiyang Zhao, Tao Jiang, Nils Morozs, David Grace, and Tim Clarke. Transfer learning: A paradigm for dynamic spectrum and topology management in flexible architectures. In *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*, pages 1–5. IEEE, 2013.

[188] Qiyang Zhao, David Grace, and Tim Clarke. Transfer learning and cooperation management: balancing the quality of service and information exchange overhead in cognitive radio networks. *Transactions on Emerging Telecommunications Technologies*, 26(2):290–301, 2015.

[189] Rongpeng Li, Zhifeng Zhao, Xianfu Chen, Jacques Palicot, and Honggang Zhang. TACT: a transfer actor-critic learning framework for energy saving in cellular radio access networks. *Wireless Communications, IEEE Transactions on*, 13(4):2000–2011, 2014.

[190] Ana Galindo-Serrano and Lorenza Giupponi. Distributed Q-learning for aggregated interference control in cognitive radio networks. *Vehicular Technology, IEEE Transactions on*, 59(4):1823–1834, 2010.

[191] Ali Imran and Ahmed Zoha. Challenges in 5G: how to empower SON with big data for enabling 5G. *Network, IEEE*, 28(6):27–33, 2014.

[192] Apache Hadoop. Apache Hadoop. http://hadoop.apache.org, 2011.

[193] Nathan Marz. *Twitter's Storm: Distributed Real-Time Computation System. The Apache Software Foundation, 2011.* 2014.

[194] Apache Storm. Storm, Distributed and Fault-Tolerant Real-time Computation. http://storm.apache.org, 2014.

[195] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, volume 10, page 10, 2010.

[196] Apache Spark. Apache spark–lightning-fast cluster computing. http://spark.apache.org, 2014.

[197] China Mobile. C-RAN: the road towards green RAN. *White Paper, ver*, 2, 2011.

[198] Ejder Bastug, Mehdi Bennis, and Mérouane Debbah. Living on the edge: The role of proactive caching in 5g wireless networks. *Communications Magazine, IEEE*, 52(8):82–89, 2014.

Appendix A: Derivation of Backpropagation Formula

This section briefly presents the derivation of the Backpropagation algorithm in adjusting the weights of a neural network. Here, we consider the hidden neuron z_j, identified in green as shown in Figure 2.2. Assuming a general K–RAT classification problem, we can formulate the error at z_j, which is the difference between the output y_n of Equation 2.3 and the Backpropagation value, as follows. Through Chain Rule [Leibniz, 1676 and L'Hôpital, 1696], we can evaluate the partial derivatives,

$$\frac{\partial E_n}{\partial w_{ij}} = \frac{\partial E_n}{\partial a_i} \frac{\partial a_i}{\partial w_{ij}}. \tag{1}$$

Let, the symbol δ be defined as the *error* as follows,

$$\delta_j \equiv \frac{\partial E_n}{\partial a_i}. \tag{2}$$

Now, using Equation 2.2, we can write,

$$\frac{\partial a_i}{\partial w_{ij}} = z_j. \tag{3}$$

Substituting Equations 3 and 2 into Equation 1, we get,

$$\frac{\partial E}{\partial w_{ij}} = \delta_j z_j, \tag{4}$$

which simplifies and generalizes Backpropagation the error from any layer $l+1$ to any node j in the previous layer l.

Referring back to Equation 2, we can use Chain Rule like before to derive the partial derivative formula,

$$\delta_j \equiv \frac{\partial E_n}{\partial a_i} = \sum_k \frac{\partial E_n}{\partial a_k} \frac{\partial a_k}{\partial a_j}. \tag{5}$$

Finally, substituting Equation 2 into 5, and making use of 2.3 and 2.4, we obtain the *Backpropagation formula*,

$$\delta_j = y_n - \hat{y}_n = g'(a_j) \sum_{k=1}^{K} w_{jk} \delta_k. \tag{6}$$

Time Domain Sample

Figures 3,4 and 5 present a time domain sample of the three radio access technologies considered in this thesis. Each subplot of the figures represent a version of the original signal ranging from 20 dB to -40 dB of Signal to Noise Ratio (SNR). In order to achive this we passed the samples through an Additive White Gaussian Noise (AWGN) channel with varying noise levels such that at the output of the channel the SNR of the signal is lowered 5 dB at a time.

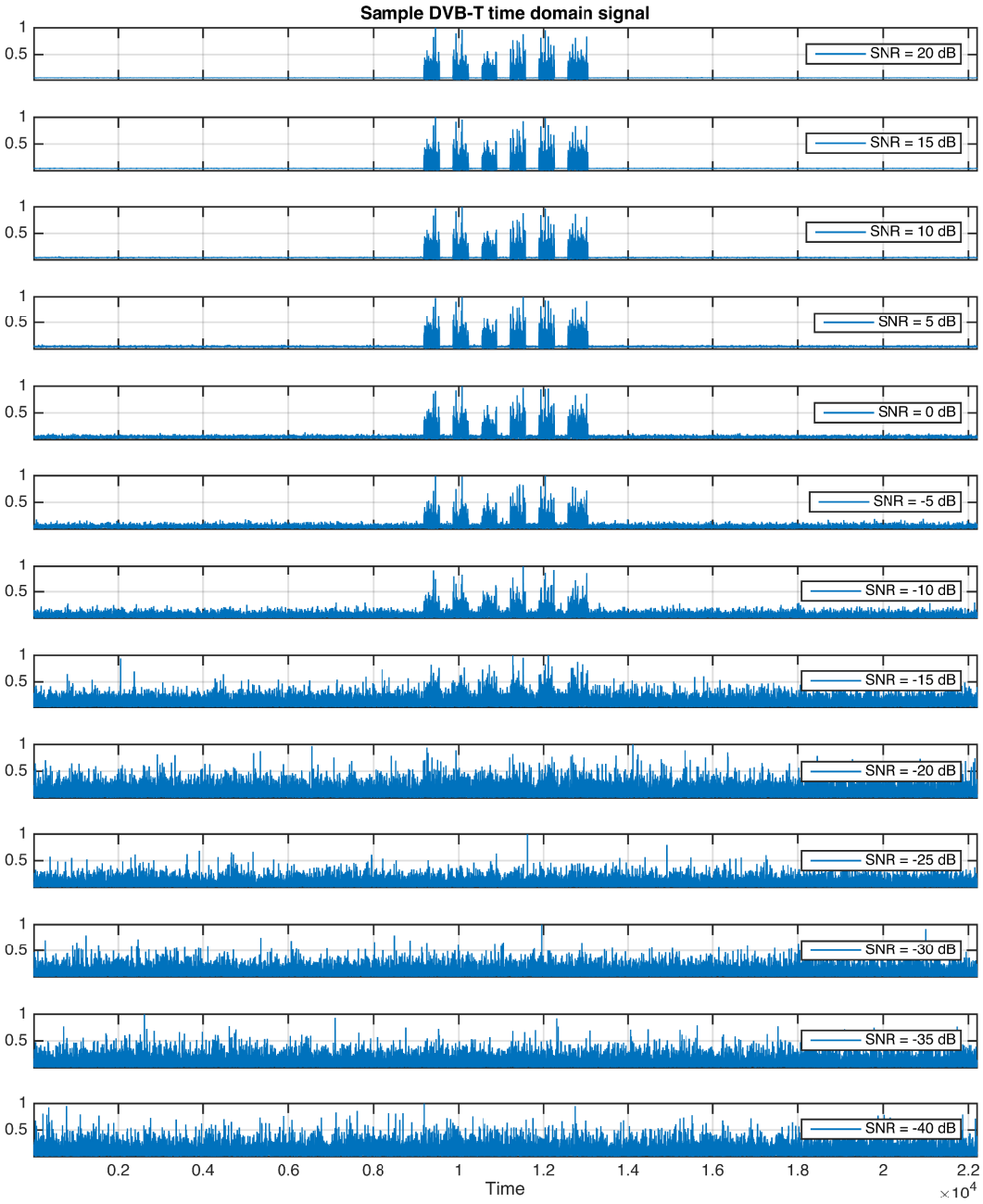


Fig. 3 A Sample DVB-T signal in Time Domain ranging from 20 dB to -40 dB

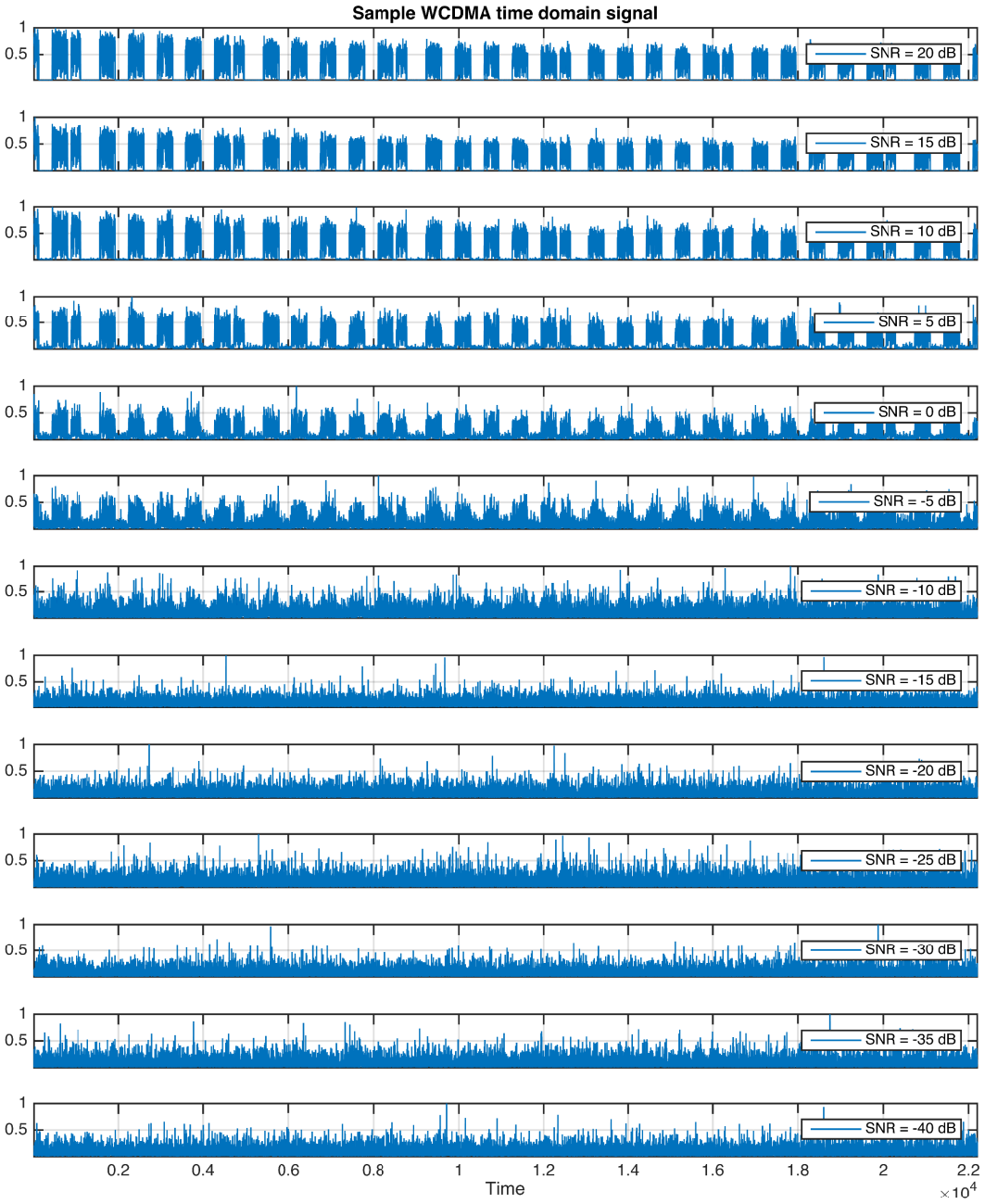


Fig. 4 A Sample WCDMA signal in Time Domain ranging from 20 dB to -40 dB

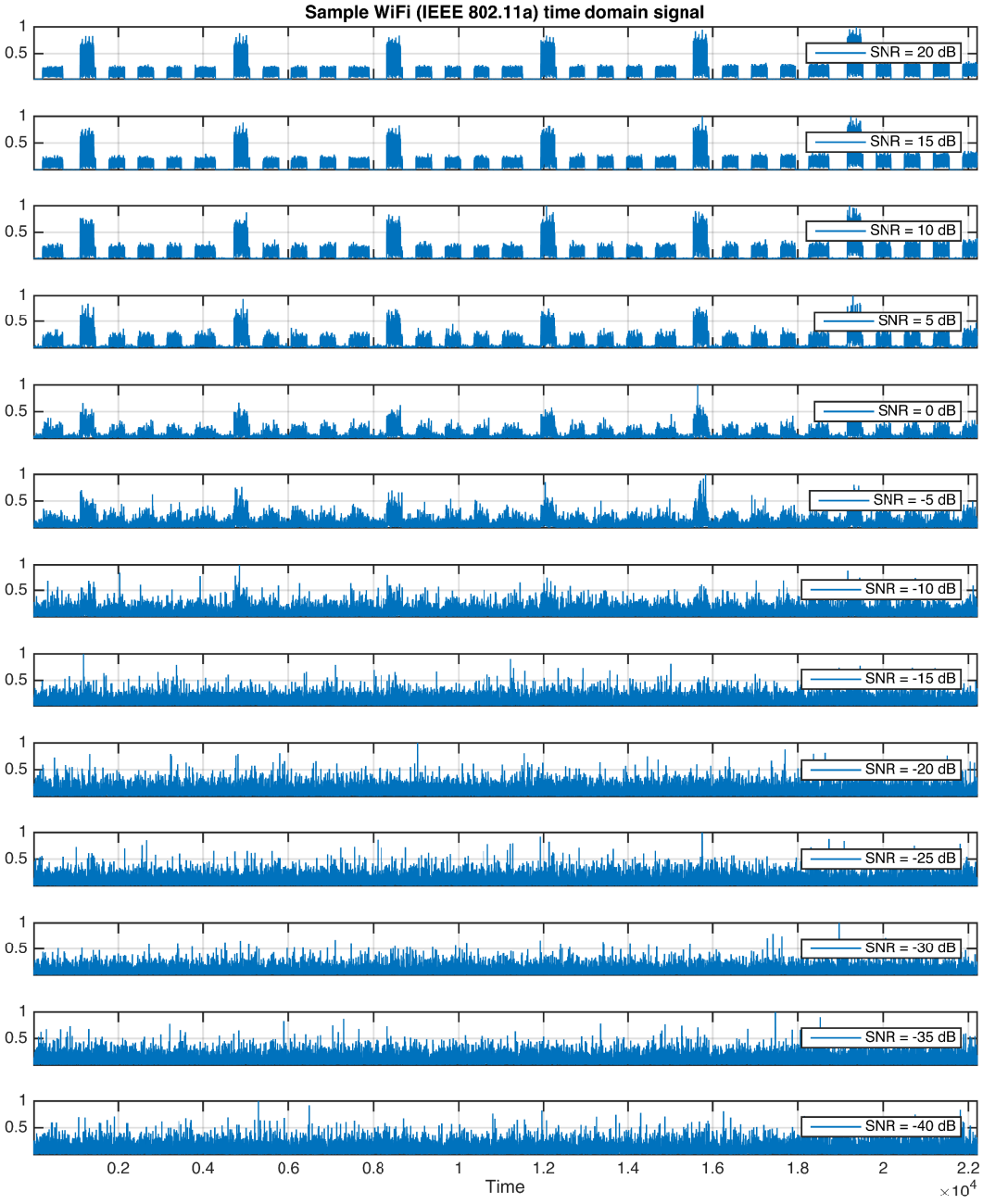


Fig. 5 A Sample of the WiFi (IEEE 802.11a) signal in Time Domain ranging from 20 dB to -40 dB

Appendix B: Boxplot Anatomy

Since boxplots are often used to display results in this thesis, here we give a brief description of its anatomy, as illustrated in Figure 6. each box represents a series of data points, where each data point could be the outcome of an experiment. The red mid line indicates the median. The lines with T-shaped ending that protrude from the box are called whiskers, and they are depicted to represent the data samples s such that $q_3 + w(q_3 - q_1) \geq s > 75^{th}$ percentile of the top, and 25^{th} percentile $\geq s > q_3 - w(q_3 - q_1)$ of the bottom samples. The maximum length of the whiskers, $w = 1.5$ units, in figures, approximately represents $\pm 2.7\sigma$ (σ: Standard Deviation), which is equivalent to 99.3 coverage[2] in a Gaussian distribution. The red plus signs are the most extreme data points, or, in other words, *outliers*.

[2]percentage of area under a distribution

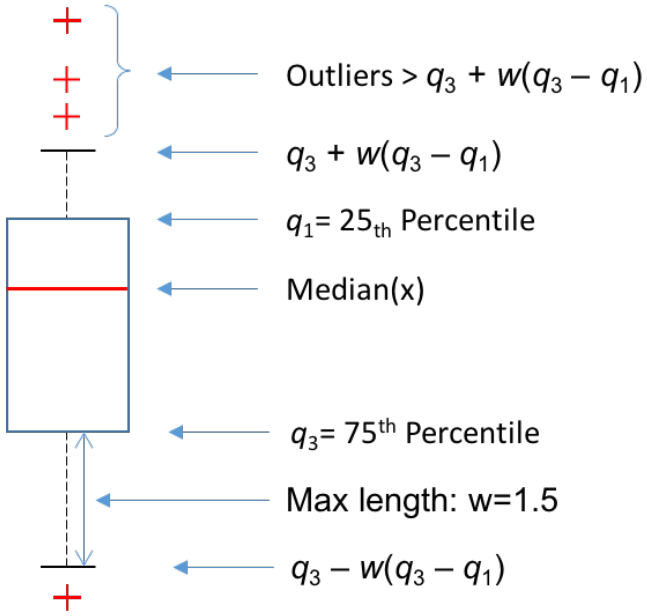


Fig. 6 Anatomy of Boxplot